



2

Milstar Radome NDI Trainer Prototype Description

By

D. J. Phair
J. A. Wilson

July 1993

Prepared for

Program Director
MILSATCOM Terminal Programs
Electronic Systems Center
Air Force Materiel Command
United States Air Force
Hanscom Air Force Base, Massachusetts



For color reproduction, please contact the product manager. All other reproductions will be in black and white.

93-27408



Approved for public release;
distribution unlimited.

Project No. 639A
Prepared by
The MITRE Corporation
Bedford, Massachusetts
Contract No. F19628-89-C-0001

When U.S. Government drawings, specifications or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.

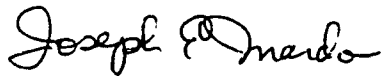


PATRICK D. MORONEY, CAPTAIN, USAF
Airborne Installation Manager



JOHN H. PULSIPHER, MAJOR, USAF
Director, Test and Deployment

FOR THE COMMANDER



JOSEPH E. MARDO, GM-15
Program Director
MILSATCOM Terminal Programs

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF COLOR PAGES WHICH DO NOT REPRODUCE LEGIBLY ON BLACK AND WHITE MICROFICHE.

REPORT DOCUMENTATION PAGE

Form Approved

OMB No 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE July 1993	3. REPORT TYPE AND DATES COVERED Final	
4. TITLE AND SUBTITLE Milstar Radome NDI Trainer Prototype Description			5. FUNDING NUMBERS F19628-89-C-0001 639A	
6. AUTHOR(S) Phair, Douglas J. Wilson, John A.				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) The MITRE Corporation 202 Burlington Road Bedford, MA 01730-1420			8. PERFORMING ORGANIZATION REPORT NUMBER MTR 92B0000085	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) Program Director, MILSATCOM Terminal Programs (ESC/MSI) Electronic Systems Center, AFMC Hanscom AFB, MA 01731-3010			10. SPONSORING/MONITORING AGENCY REPORT NUMBER ESC-TR-93-158	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) Radomes constructed from a Kevlar/polyester composite are being produced for U.S. Air Force Milstar terminals for use on airborne platforms. The MITRE Corporation is prototyping a computer-based training system that will train nondestructive inspection (NDI) personnel how to inspect Milstar radomes using ultrasonics. The Trainer uses state-of-the-art multimedia tools to integrate ultrasonic signals, simulated test equipment, microscope images, digitized video, talking help agents, and a database of radomes with configurable flaws. The resultant training environment is both visual and highly interactive. This document describes the Trainer prototype in terms of its functions, design architecture, and development tools.				
14. SUBJECT TERMS Computer-based Training Milstar Radome Multimedia Training			15. NUMBER OF PAGES 64	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT SAR	

ACKNOWLEDGMENTS

This document has been prepared by The MITRE Corporation under Project 639A, Contract No. F19628-89-C-0001. The contract is sponsored by the Electronic Systems Center, Air Force Materiel Command, United States Air Force, Hanscom Air Force Base, Massachusetts 01731-3010.

The authors are grateful for the assistance of several individuals. Dick Lamontagne contributed to the initial phases of the prototyping. Dick provided background information, initial design ideas, and helped get the effort started. We are also grateful to the nondestructive inspection (NDI) personnel at Sacramento Air Logistics Center, including Danny Anderson, Don Bailey, Donna Ballard, and Al Rogel. Their comments improved our understanding of the application domain and also helped us refine our prototyping to meet the training requirements of nondestructive inspections. Don Bailey deserves special thanks for giving extra time and taking a personal interest in this application. Don's vision, inputs, and personal commitment to this effort have helped immeasurably and are reflected in many areas of the prototype. We would also like to acknowledge the review and support of our ESC Program Office and MITRE management team: Capt. P. M. Loughnane, Maj. T. R. Whitacre, M. E. Fitzgerald, S. M. Jolly, L. S. Metzger, P. J. Nesky, D. P. White, and W. C. Wilder. Their collective assistance helped the Trainer in many ways. Last but not least, we are grateful to Roberta A. Carrara (J103). Her review greatly improved this document.

DTIC QUALITY INSPECTED 4

Accession For	
NTIS CRARI	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution	
Announcements	
Periodicals	
Dist	
A-1	

TABLE OF CONTENTS

SECTION	PAGE
1 Introduction	1
1.1 Overview of Project	1
1.2 Purpose and Scope of Document	1
1.3 Organization of Document	2
2 Background	3
2.1 Current Radome Inspection Process	3
2.2 Prototyping Objectives	5
3 Functional Description	7
3.1 Display Organization	7
3.2 User Threads	8
3.2.1 Inspection Training	8
3.2.2 Radome Editing	24
3.2.3 Trainer Configuration	30
3.3 System Requirements and Installation	32
4 Development Approach	34
4.1 Evolutionary Development Plan	34
4.2 Architectural Design	37
4.3 Development Tools and Utilities	41
4.3.1 Hypermedia Authoring Environment	42
4.3.2 External Command Compilers	43
4.3.3 Multimedia Development Tools	45
5 Other Applications	50
6 Conclusion	52
List of References	53
Glossary	54

LIST OF FIGURES

FIGURE	PAGE
1 Pulse-Echo Ultrasonic Inspection of Milstar Radome	4
2 Display and Function Hierarchy	7
3 Login Completed and Main Menu Presented	9
4 Calibration Display with Video Tutorial	10
5 Adjusting the Simulated Ultrasonic Meter	11
6 Applying Couplant to the Calibration Block	13
7 Ultrasonic Meter Calibrated	14
8 Grid Lines Applied to Radome Prior to Inspection	15
9 Inspection Display with Couplant Being Applied	16
10 On-Line Help Agent Assists Student	17
11 Digitized Video Tutorial Illustrates Proper Scanning Techniques	18
12 Student Inspects Section of Radome	19
13 Grease Pencil Selected after Flaw Has Been Located	21
14 Flaw Outlined and Result Entered in Inspection Notebook	22
15 Signal and Image Tutorial To Illustrate Signal Types	23
16 Microscope Image Magnified in Tutorial	24
17 Instructor Examines Flaw Areas in Radome	25
18 Instructor Reviews Student Inspection Results	26
19 Instructor Edits Radome in Database	27
20 Instructor Selects Flaw Type To Add to Radome	28
21 Instructor Lists User Accounts	29
22 Instructor Reviews Couplant Parameter Settings	30
23 Designer Activates A-Scan Database Utility	31
24 Designer Adds New A-Scan to Database	32
25 Evolutionary Rapid Prototyping	36
26 System Architecture	38
27 COTS Interfaces	39
28 Database Structure	40

FIGURE	PAGE
29 COTS-Based Architecture	41
30 Sample SuperCard Display	42
31 <i>Compileit!</i> Development Environment	44
32 <i>Compileit!</i> Debugger	45
33 Sample MacRecorder Display	46
34 InterFACE Agent Editor	47
35 <i>MediaGrabber</i> Development Environment	49

SECTION 1

INTRODUCTION

1.1 OVERVIEW OF PROJECT

Radomes constructed from a Kevlar/polyester composite are being produced for the U. S. Air Force Milstar Terminals for use on airborne platforms. The radomes will be inspected in the field with hand-held ultrasonic test equipment. The correlation of ultrasonic signals to specific flaw types in this new composite requires careful manual interpretation. Consistent logging of defects and proper application of inspection procedures are also important to allow the results from different inspections to be compared.

The MITRE Corporation is prototyping a computer-based training system that will train nondestructive inspection (NDI) personnel how to inspect Milstar radomes using ultrasonics. The Radome Inspection Trainer, henceforth referred to as the Trainer, consists of a Macintosh computer system, multimedia software tools, and software developed in an object-based programming language. This work is being performed for the Air Force Electronic Systems Division (ESD) and coordinated closely with NDI personnel from Sacramento Air Logistics Center (SM-ALC).

Most NDI training is conducted with written procedures, real parts, and test equipment. Computer-based training in NDI has been limited to question-and-answer programs that do not capture the inherently visual aspects of NDI. The Trainer capitalizes on state-of-the-art multimedia tools for composite images, digitized video, test sample (radome) graphics, still and simulated signals, talking agents, and other voice. This experimental application has successfully shown how multimedia technology can provide highly realistic, interactive, and cost-effective training environments for NDI training.

1.2 PURPOSE AND SCOPE OF DOCUMENT

This document describes the displays, functions, and multimedia features of the Milstar Radome Inspection Trainer Prototype. This description covers the use of the Trainer by three classes of users: students, trained inspectors, and designers. Display hard copies illustrate user threads through the Trainer. The document also briefly describes the evolutionary development plan, architectural design, and commercial tools and utilities used to accelerate the prototyping.

The purpose of the document is to provide a functional, easily readable description of the Trainer. The document is not intended to serve as a user's manual nor is it intended to describe the detailed design and implementation. Both topics will be presented in depth in future technical reports. We make no assumptions about the reader's background.

1.3 ORGANIZATION OF DOCUMENT

Section 2 describes the current inspection process for Milstar radomes and highlights the areas that require considerable expertise and training. The prototyping objectives for the Trainer are also presented in section 2. Section 3 outlines the display organization and presents functional threads for student, instructor, and design users. In section 3, we also discuss system execution and installation. In section 4, the development plan, architectural design, and development tools and utilities are briefly described. Section 5 includes other areas in nondestructive evaluation that could benefit from the applications of multimedia computer technology. Finally, we present our conclusion in section 6.

SECTION 2

BACKGROUND

2.1 CURRENT RADOME INSPECTION PROCESS

For the purpose of the Trainer, the Milstar radome refers specifically to the Milstar Airborne Command Post EC-135 antenna radome. The radome is approximately 18 feet long and 6 feet across at its widest point. In addition to being a large structure to inspect, the radome has an unusual shape. The radome is constructed out of a Kevlar/polyester composite (15 plies) and is only 0.15 inches thick. The composite has been selected to meet the stringent transmission efficiency requirements for Milstar. Experience with this new composite is very limited; therefore, structural integrity is a particularly significant concern. Inspections on the initial radome, being conducted by NDI scientists from SM-ALC, occur frequently.

Inspections on the Milstar radome are performed using pulse-echo ultrasonics. In this NDI method, ultrasonic energy is transmitted between a transducer and the radome through a coupling medium, such as water or gel. This process is illustrated in figure 1. Couplant excludes the air interface between the transducer and the radome. Ultrasound waves are propagated through the radome, and the reflected energy from the front and back surfaces are received by the transducer and displayed on an ultrasonic meter as amplitude over time. This presentation is called an A-scan and is used in most field applications. Notice that figure 1 shows the front surface echo (FSE) and back surface echo (BSE) of the radome. Any flaws (or discontinuities) in the Kevlar/polyester composite will be shown as reflected echoes in the A-scan between the front and back surface echoes. In addition, reflected energy from flaws in the radome will result in loss of the BSE. Consequently, loss of BSE is the primary criterion for detecting flaws.

The A-scan also contains distance information about where the flaw is located in the radome material. When a flaw echo is shown on the ultrasonic meter, its position relative to the front and back surface echoes indicate how far into the material the flaw is located. An echo close to the FSE indicates a flaw near the surface; an echo near the BSE indicates a flaw near the inside of the radome. A signal with negligible echoes between the FSE and BSE indicates that there are no flaws at that location. Figure 1 shows a scanning electron microscope (SEM) image of a flawed area of a radome. Notice the positions of the echoes in relation to the actual positions of the front surface, flaw (discontinuity), and back surface. Further information on ultrasonic inspection and other forms of NDI is available in [1].

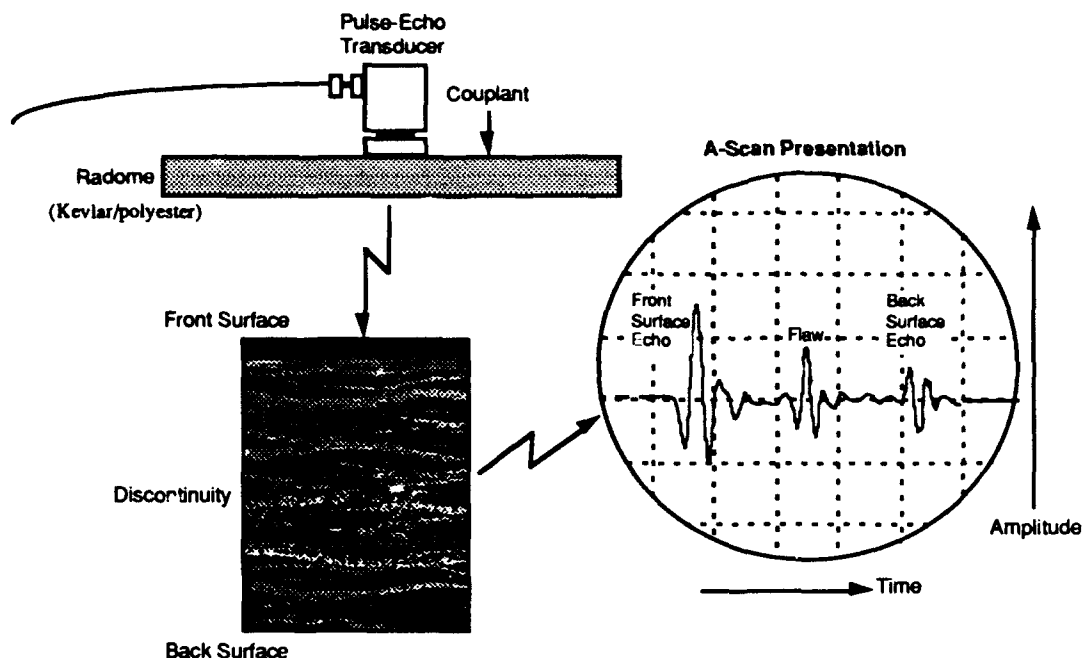


Figure 1. Pulse-Echo Ultrasonic Inspection of Milstar Radome

The inspection procedure requires the following equipment: a portable ultrasonic instrument; a 5-MHz, 0.5-inch diameter transducer; a 0.5-inch long, Lucite ultrasonic delay line; a IIW Type 2, aluminum ultrasonic reference block; water, oil, or gel for ultrasonic couplant; and a 6-foot minimum length coaxial cable. The inspection procedure also requires specific steps that must be followed to properly calibrate the ultrasonic equipment and perform the inspection. These steps will be outlined in paragraph 3.2, in order to illustrate how the Trainer closely simulates real inspections. The Trainer displays shown in paragraph 3.2 will also serve to explain the different inspection and calibration steps. To explain the training challenges for this inspection, a general description is given below.

Calibration of the ultrasonic equipment is a critical first step to performing inspection. If the calibration is performed incorrectly, the results of the entire inspection may be incorrect. Calibration of the test equipment is performed using an aluminum block (as opposed to a piece of the composite) because of concerns that the Kevlar/composite's ultrasonic characteristics may change over time. The inspector will adjust the controls of the ultrasonic meter by using the test block. The control settings are critical. For example, the inspection is particularly sensitive to the frequency setting. Higher frequencies will provide greater sensitivity for detecting flaws but require greater power to penetrate the composite material (frequencies between 2.25 and 10 MHz are typically used in pulse-echo ultrasonic applications). Also, since higher frequencies are more sensitive, the resultant signal can interfere with

discriminating between nuances in the composite and relevant flaws. For this particular composite, a frequency of 5 MHz has been recommended for detecting flaws as small as 0.1 square inch.

After he has calibrated the ultrasonic test equipment, the inspector will mark the radome in one-square-foot sections and inspect one section at a time. Inspections by experienced NDI personnel from SM-ALC take 20 to 40 hours. The lengthy inspection is a result of the radome's large surface area. Furthermore, it is difficult to watch the ultrasonic meter for signal changes and move the transducer in a precise scanning motion so that none of the surface is missed. Flaws in the composite can include, but are not limited to, manufacturing voids (porosity), foreign material inclusions, delaminations, and impact damage. Some flaws have fairly obvious signal changes; other flaws have very subtle changes and will look like a normal signal. The inspector must carefully record the results so that small flaws can be tracked over time, from inspection to inspection. The inspection is manually intensive, requires considerable expertise, and is not covered in existing NDI training procedures. The Trainer will allow a new inspector to practice this inspection before inspecting an actual radome. The specific objectives and requirements for this work are outlined below.

2.2 PROTOTYPING OBJECTIVES

The general mission of the Trainer is to improve the reliability of Milstar radome field inspections, entailing a number of functional objectives that address the required capabilities of the Trainer. An additional objective is to reduce development costs and provide an evolving capability that NDI experts and potential users can evaluate.

First, the Trainer shall help new inspectors learn how to correlate ultrasonic signals to particular defect types. Signal recognition requires considerable expertise and correct classification of defect areas is critical. Because ultrasonics is signal-based, the relationship between actual waveforms and physical properties in a test sample is somewhat abstract. Moreover, subtle differences between waveforms can take time to learn. The Trainer will store digitally acquired ultrasonic waveforms from actual field inspections so that students can practice with real data. The library of ultrasonic signals stored in the Trainer will also serve as a reference standard for defect classification.

Second, the Trainer shall facilitate consistent logging of defects. An inspector, upon detecting a new defect in the Trainer, must enter specific attributes to describe the defect. The attributes include the type, size, and location of the defect. Consistent training will improve the chance that field inspectors enter the same data when logging defects. Careful recording will also permit the results of one inspection to be compared to subsequent inspections so that defect growth can be tracked. Many defects will not ground the aircraft but must be watched carefully for changes.

Third, the Trainer shall provide practice on various aspects of the inspection process, such as calibration and scanning with the transducer. Field inspections can easily be flawed by incorrect calibration or imprecise scanning. Multimedia technology can have an enormous benefit in this area over conventional teaching materials; typical training methods do not emphasize 'hands-on' practice. Inspection instruments, such as an ultrasonic meter, can be

simulated to interact with digitally stored radomes. Graphical images of radomes can be manipulated in real time by the student user. This same approach is not peculiar to radomes; it can be readily applied to other test parts and inspection problems. In short, multimedia provides an inherently visual training and educational environment that cannot be provided using conventional training methods.

Fourth, the Trainer shall provide on-line help and tutorials for novice students. Once again, multimedia technology offers an inherently visual framework for introducing new information. The Trainer shall use sound, voice, imagery, and video to provide on-line help that is both informative and engaging. Furthermore, the Trainer must assume minimal computer expertise since many users will have only limited computer experience. As such, a point-and-click interface is provided that minimizes keying in information and the chance of user input errors.

Last, developing the Trainer shall require only minimal resources. Although custom software is being developed, it is being carried out by a single engineer and accelerated through a technique called rapid prototyping. To help satisfy this development objective, the Trainer will capitalize on commercially available, state-of-the-art multimedia and software tools. The commercial tools allow highly functional prototypes with sophisticated user interfaces to be built quickly while minimizing new software development. Costs for the hardware and software are also small -- development is occurring on a Macintosh computer system using readily available development tools. We describe the development in more detail in section 4.

SECTION 3

FUNCTIONAL DESCRIPTION

3.1 DISPLAY ORGANIZATION

The Trainer has been prototyped to address the key requirements of three types of users: students, instructors, and designers. Consequently, the organization of displays and functions is divided into three areas, as illustrated in figure 2.

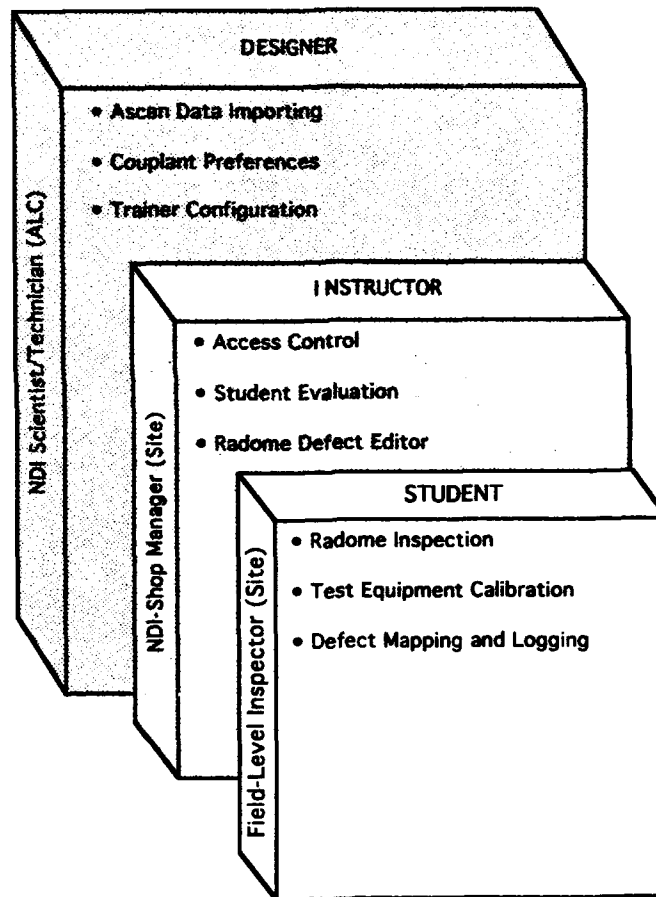


Figure 2. Display and Function Hierarchy

The most important user of the application is the student, and most of the Trainer addresses student training functions. We assume that a student is probably a field-level NDI inspector and has previously completed basic NDI training, such as the Air Force NDI Technical School, and is familiar with ultrasonics. The second type of user is the instructor. We expect that the instructor is either the individual that runs the NDI shop at a field-level site or is a senior inspector. The last type of user is a designer. This individual probably resides at an Air Logistics Center (ALC) and is a senior NDI scientist or technician. For this application, the inspectors at SM-ALC are considered designers. Designers must be intimately familiar with the Kevlar/polyester composite and the associated ultrasonic signals so they can decide when new signals should be added to update the Trainer. Figure 2 shows the functional areas for each of the three types of users.

3.2 USER THREADS

The capabilities of the Trainer can best be illustrated by examining representative threads that pertain to student, instructor, and designer use. For the student, we will describe the displays and functions used in a typical training session. This includes calibration, on-line help, and radome inspection. For the instructor, the radome editing capabilities will be described. These functions allow instructors to create custom radomes with flaws that reference the ultrasonic database. An instructor can create a radome as easy or difficult as is appropriate. The Trainer software will automatically assign the skill level (novice, intermediate, expert) required for a new radome. The third type of user is the designer. A designer can perform functions that tailor the operation of the Trainer, such as loading new ultrasonic signal data. The three types of user, student, instructor, and designer, also represent levels of access into the Trainer. A designer can access all functions, an instructor can access any function except designer functions, and students can only access student functions.

3.2.1 Inspection Training

The majority of the capabilities in the Trainer are for student users. The first step a student will take in operating the Trainer is to log in to his account. Creating user accounts is a function that can only be performed by instructors and designers and will be described later. The login procedure is straightforward: the student simply enters name and password. The system will inform the student of the last date and time of connection. As we will see later, student accounts are important because they contain any inspection results that the student has recorded while training. This allows an instructor to evaluate a student's inspection results and permits the student to perform an inspection over several runs by saving and recalling his work. As mentioned earlier, actual inspections take 20 to 40 hours. We anticipate an inspection of a simulated radome (only one side) will require at least a couple of hours.

Once the student has logged in, the main menu bar is displayed (figure 3). At this point, the student can request help (e.g., view a video tutorial, perform calibration, inspect a radome, change his password, or quit the Trainer (logout)). Typically, a student will perform calibration. It is important to note that the student should not begin an inspection at this point, since the Trainer assumes the ultrasonic equipment is not calibrated at the start of each run. If the student does decide to inspect the radome, the results will most likely be incorrect.

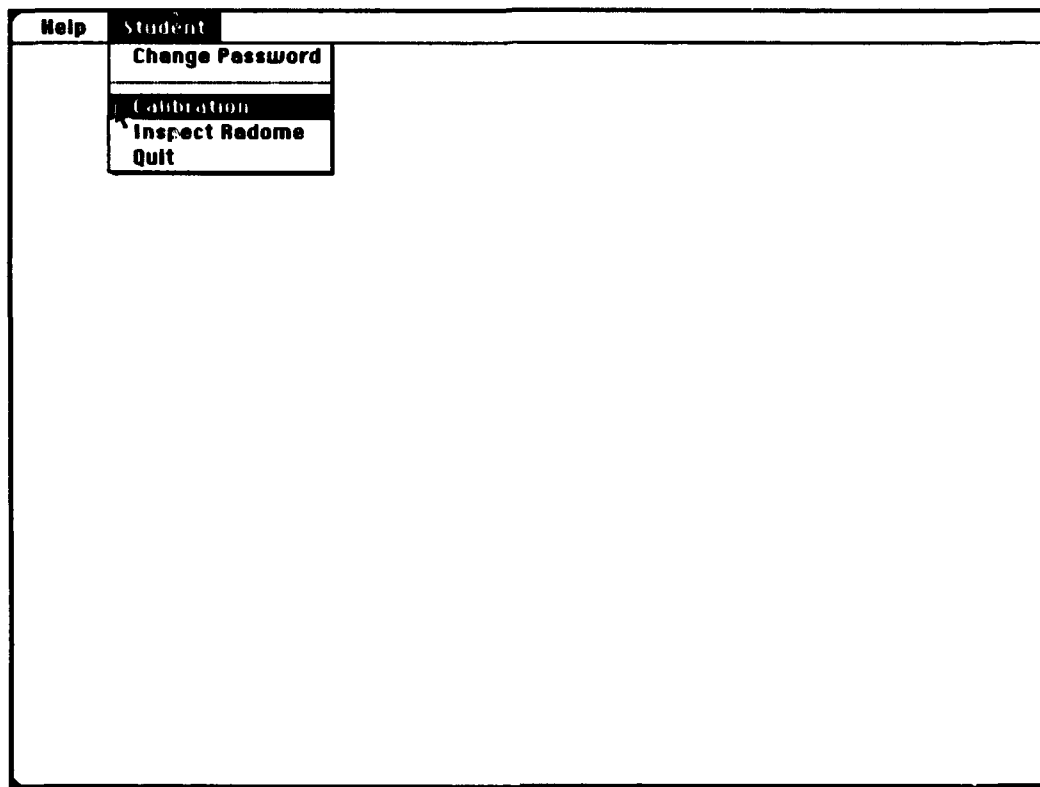


Figure 3. Login Completed and Main Menu Presented

Once the calibration function is selected, the display appears (figure 4). The student interacts with several areas of the calibration display. The menu bar at the top of the display allows other functions to be performed and the student to navigate to other areas of the Trainer. The IIW Type 2, aluminum ultrasonic reference block is shown as a graphic image in the display. Also, a simulated ultrasonic meter is shown and provides the controls found on most ultrasonic test equipment. On the left side of the display, a small palette contains calibration tools that the student will work with. The top, rectangular tool is the ultrasonic transducer. The second tool is a couplant brush. The third tool, an arrow, is the selection tool, which works with the menus and the ultrasonic meter. Each of these tools and areas of the display will be described in further detail in the paragraphs below. Finally, a window in the upper-right corner of the display shows a video tutorial on calibration. The student selects this tutorial from the help menu. The video tutorial can be activated or closed at any time as the student learns the calibration process.

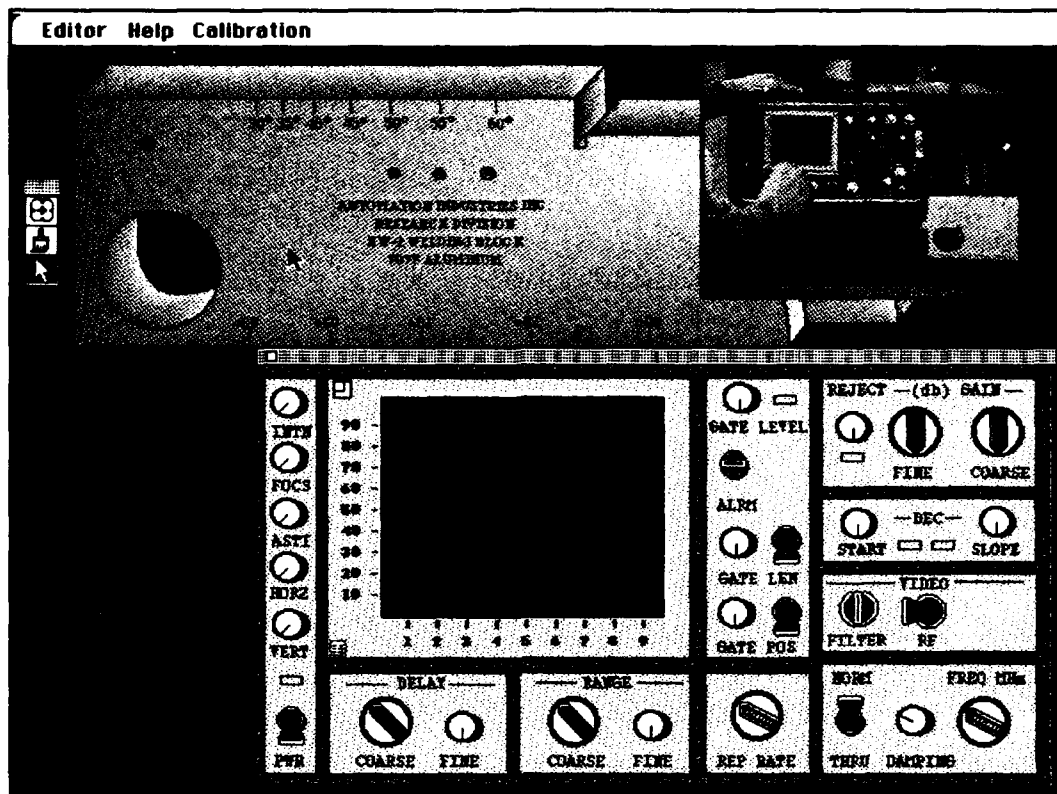


Figure 4. Calibration Display with Video Tutorial

The use of digitized video tutorials is an important example of how multimedia computer technology can enhance NDI training. Calibration of the ultrasonic meter is absolutely critical to the integrity of the inspection. In particular, calibration requires careful setting of several controls, such as frequency, gain, range, and delay. If the student performs calibration incorrectly, the entire inspection results may be fallible. The video tutorial shows how each control operates and the relationship between different controls on the meter. The tutorials (video frames) are stored (and compressed) digitally on hard disk, so playing the tutorials requires no external devices and occurs transparently to the user. A separate control panel can be selected that allows rewind, fast forward, and other typical movie-viewing functions. The control panel is not shown in figure 4.

To calibrate the simulated ultrasonic test equipment, the student must place the different controls in the correct positions. Initially, when the student selects the calibration function in the Trainer, the ultrasonic meter is automatically set to an unknown (and incorrect) setting. This ensures that the student must work at configuring the meter correctly.

In figure 5, the student has turned on the meter, selected the transducer from the tool palette and placed it on the block, and applied some gain. The meter shows the initial ring and FSE off the calibration block. Without any couplant, the ultrasound cannot penetrate the block for the BSE. In order for the student to complete calibration, couplant must first be applied to the calibration block.

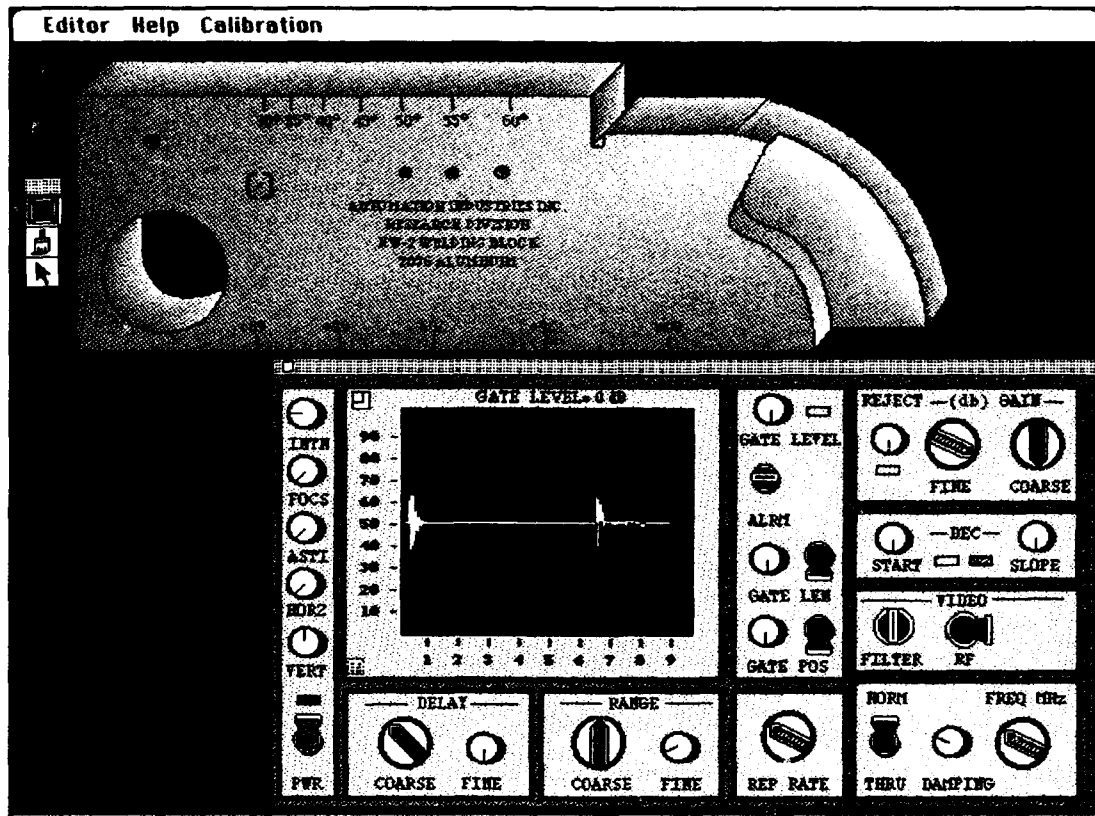


Figure 5. Adjusting the Simulated Ultrasonic Meter

It is instructive at this point to briefly describe the various controls on the simulated ultrasonic meter. The left side of the meter contains a bank of five controls and the power switch (labeled *PWR*). *INTN* adjusts the intensity of the ultrasonic signal; the *FOCUS* control adjusts the focus of the display signal. Similarly, the *ASTI* control adjusts the symmetrical alignment of the image; the *HORZ* and *VERT* controls adjust the signal's horizontal and vertical position.

Beneath the display area of the ultrasonic meter are delay and range controls, which are particularly important for the student. The *COARSE-DELAY* and *FINE-DELAY* switches provide a sweep delay from the initial ring off the interface between the block and transducer (either a ring off the couplant or ring off air when there is no couplant). When the student calibrates the equipment, he will adjust the delay controls so that the leading edge of the FSE is located at the extreme left side of the display in the ultrasonic meter. *COARSE-RANGE* and *FINE-RANGE* provide control of the signal's horizontal sweep. Adjusting the range will increase or decrease the distance between the FSE, BSE, and any other echoes that are being displayed.

Directly to the right of the meter display are controls for gate and repetition rate. *REP-RATE* simply controls the refresh rate of the signal; *GATE-LEN* and *GATE-POS* adjust the length and starting point of the gate, respectively. The controls allow triggering of amplitude readings beyond a certain tolerance within a preselected interval after the initial pulse (ring). *GATE-LEVEL* sets the level which the signal must exceed before it generates an alarm. A visual LED is provided for gate alarm. In addition, the *GATE-ALRM* control switches between audible and visual alarms and only visual alarm. For this composite, the student should set the threshold so that any BSE that falls below 50 percent FSE is alarmed.

The right side of the meter has four areas of controls. The top three controls pertain to gain. In particular, the *REJECT* knob controls rejection of low-amplitude signals on the display. The *GAIN-COARSE* and *GAIN-FINE* controls adjust the displayed signal's vertical gain. The *START-DEC* and *SLOPE-DEC* controls, not implemented for this application, are not functional. The *VIDEO-RF* switch permits the signal to be displayed as it is generated by the transducer (video) or as amplified RF. The *FILTER* switch is not implemented for this application. The last three controls, at the bottom-right corner of the meter, are *NORM-THRU*, *DAMPING*, and *FREQ MHz*. *NORM-THRU* simply selects whether the student is applying a normal or thru transmission ultrasonic inspection. This application is a normal (single transducer) form of inspection (i.e., pulse-echo). *DAMPING* controls ringing of the initial pulse. *FREQ MHz* controls the transducer frequency increments of 1, 2, 5, and 10 MHz, and broadband. Since the ultrasonic signals in the Trainer have been recorded at 5-MHz and the procedure calls for 5-MHz pulse-echo, any other frequency selection will have an adverse impact on the displayed signal.

Students need to know three other important attributes of the meter, all in the display area. In the upper-left corner of the meter display, a small symbol shows a very small square inside a slightly larger square. When the student clicks on this symbol, the meter either expands or contracts. Thus far, the figures have shown the meter in its expanded form, showing all controls. When the meter is contracted, only the display window is shown. This allows the student to conserve screen space once the meter has been adjusted, an important consideration during inspection. The second attribute of the display area is a very small symbol in the bottom-left corner. When the student clicks on this symbol, another window is displayed that shows the digital settings for each of the controls. The digital value of the last touched control is shown at the top of the display area. This is the third attribute of this area that is important to the student. The student can interrogate any control setting simply by moving his cursor over the control. It is not necessary to make a change.

In figure 6, the student has selected the couplant tool, brushed some couplant on top of the calibration block, and selected the transducer and placed it on top of the block. Note that the transducer cursor changes as the orientation is changed from the side to the top of the block. The student has also performed some adjustments to controls on the ultrasonic meter. The *COARSE-DELAY* and *FINE-DELAY* settings have been set so the initial ring (off the couplant) is not displayed. The display is now showing the FSE and the echo off the large hole in the block. The student has also adjusted the *COARSE-RANGE* and *FINE-RANGE* controls so that the horizontal sweep is increased. This allows the FSE, BSE, and any flaw echoes between to be closely examined. Notice that the top of the meter display shows the current range setting. This is triggered whenever a student manipulates a control (the hand selection cursor can be seen on the *FINE-RANGE* control).

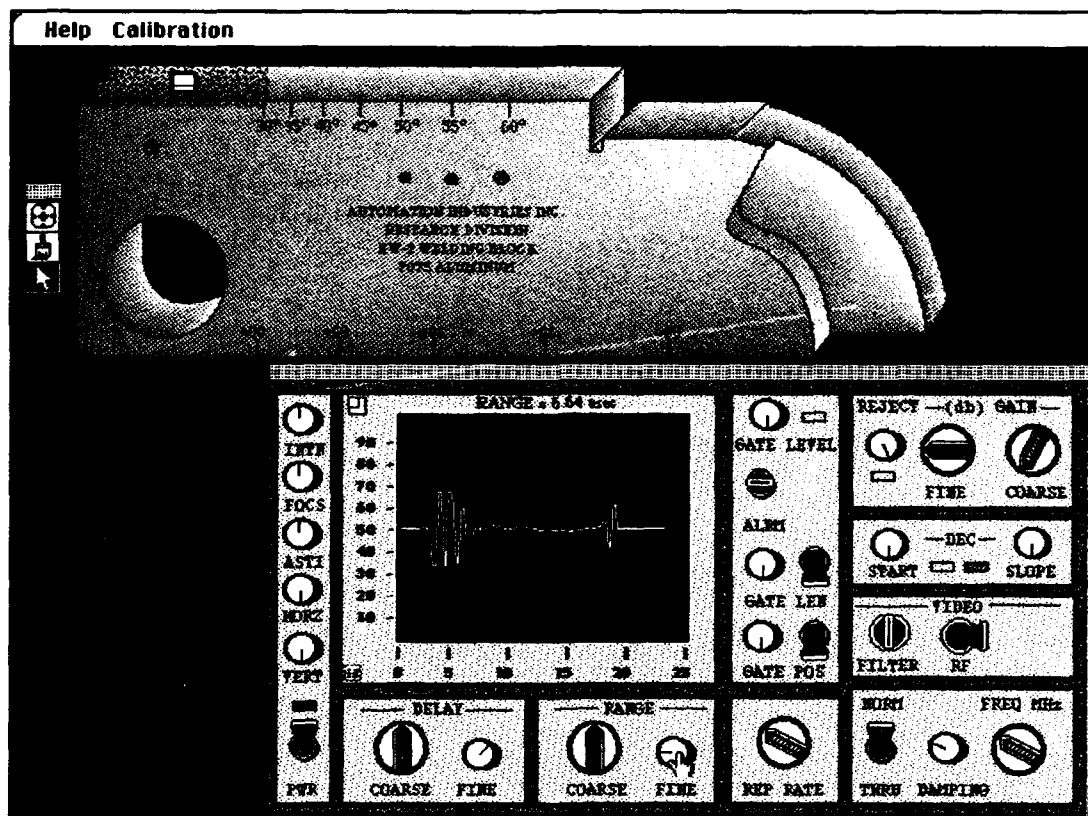


Figure 6. Applying Couplant to the Calibration Block

Figure 7 illustrates the ultrasonic meter after the student has correctly calibrated it. The three echoes shown in the meter are the FSE from the top of the calibration block, the small hole in the block, and the large hole in the block. The three echoes indicate correct calibration and result from the student correctly positioning the transducer, applying couplant, and adjusting the various controls on the meter. The student may now proceed to inspect a radome.

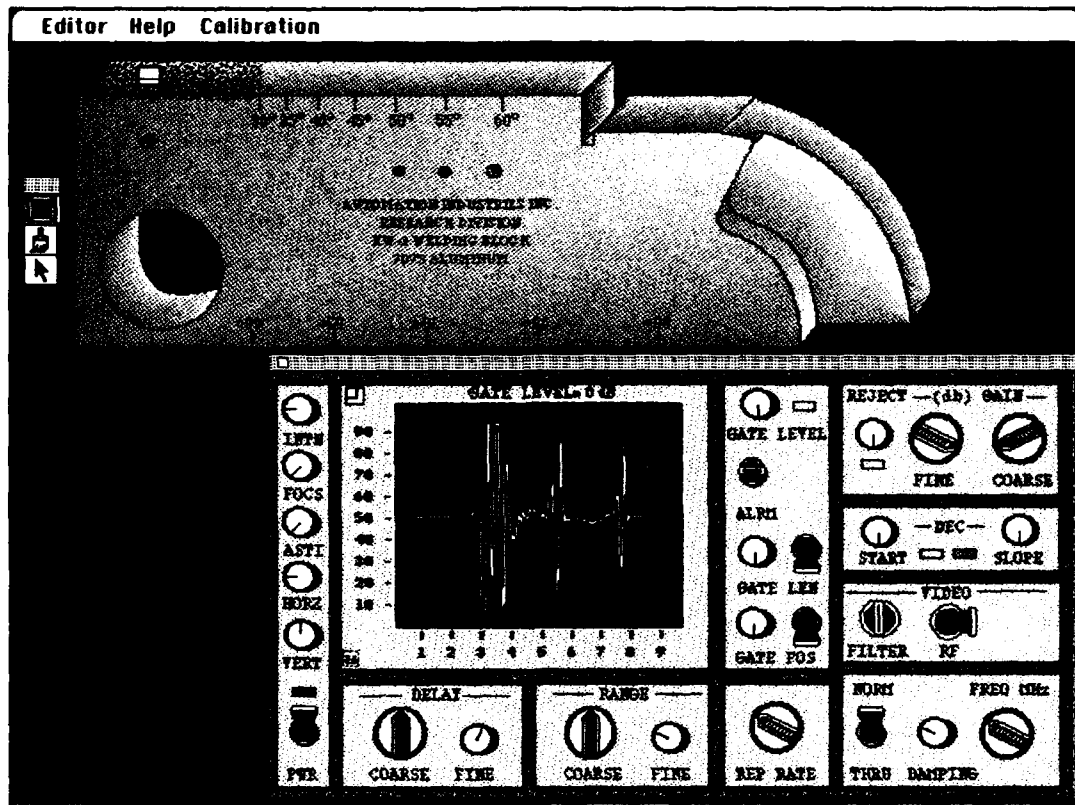


Figure 7. Ultrasonic Meter Calibrated

To begin inspecting a radome, the student (or instructor) selects a radome from the radome database that is commensurate with the student's skill level. Figure 8 shows the Milstar radome, drawn to scale, as a graphic that the student will work with. Actually, this is a radome from the database and there are flaws hidden in the material. The length of the radome, in inches, is shown. In addition, the fasteners are shown. Figure 8 also shows a grid that has been assigned to the radome. The inspection procedures indicate that the radome should be partitioned into 1-square-foot sections. The student accomplishes this in the Trainer by choosing a menu option that automatically applies the grid. The student will use this grid to

perform the inspection one section at a time. Note that only one side of the radome is shown. Since the primary objective of the Trainer is practice with the various ultrasonic signals, the decision was made that having the student scan one side of the radome in the Trainer would be sufficient. To inspect a particular section of the radome, the student simply clicks on any one of the squares in the grid. (Note: The procedure recommends that the student begin at the front of the radome and work back.)

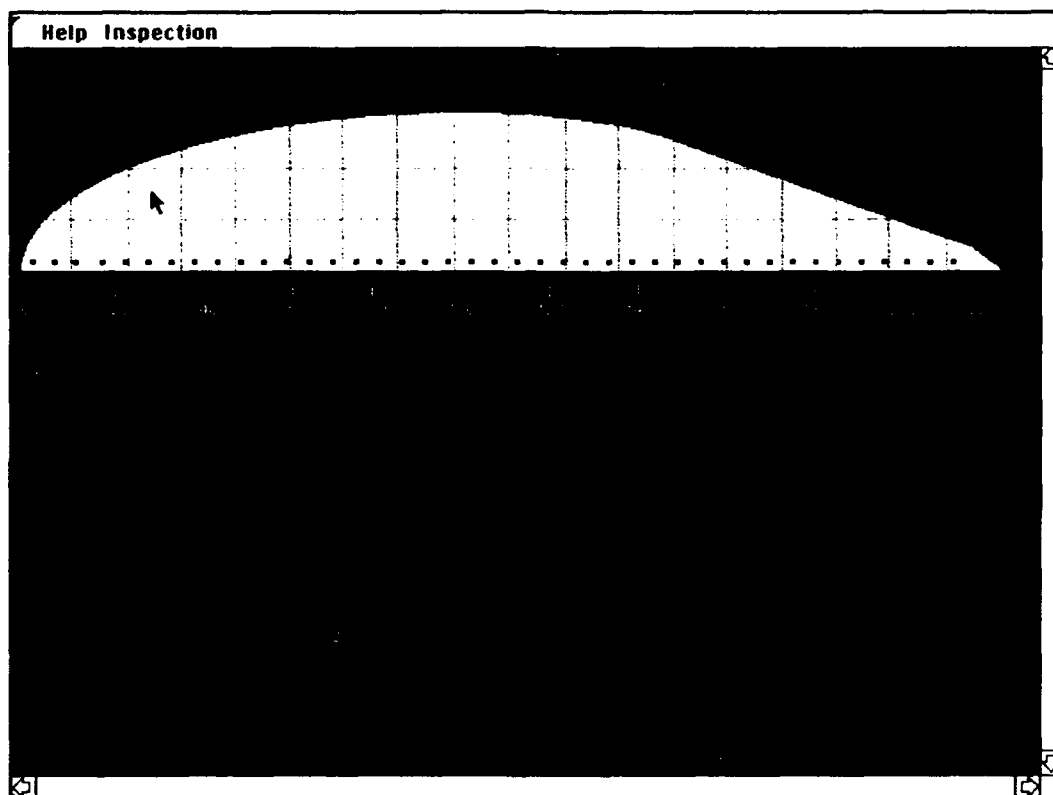


Figure 8. Grid Lines Applied to Radome Prior to Inspection

Figure 9 shows the inspection display, where a student will spend most of his time in the Trainer. This display is activated once the student has selected (clicked on and zoomed in) the section of the radome to inspect. After the student has zoomed in initially, other means for moving around the radome become available and are explained below.

A student works with several areas of the inspection display. The student may have any number of areas open at once, each of which will be briefly described. The menu bar at the top of the display allows the student to call on tutorials and on-line help, and to navigate to other areas of the Trainer. The inspection tool palette is shown on the left side of the display. The tools that are available from this palette are the ultrasonic transducer, a set of rulers, an inspection notebook, a flaw marking tool (grease pencil), a couplant paint brush or spray can, a check-mark tool (grease pencil) for indicating what areas of the radome have been inspected, and the selection (arrow) tool for manipulating the ultrasonic meter and making menu selections. The simulated ultrasonic meter shown in the bottom-right corner of this display is selected to show only the ultrasonic signal. The user can enlarge the meter view to access the various controls on the front panel (using the small square symbol in the top-left corner as described earlier). Hiding the controls permits more of the radome surface to be viewed on the display, and besides, the controls will typically not be changed after calibration (with exception of the edgeband). Horizontal and vertical scroll bars along the boundary of the display allow the user to scroll to other areas of the radome. Other functions are available on this display; however, the student will start with the meter and tool palette.

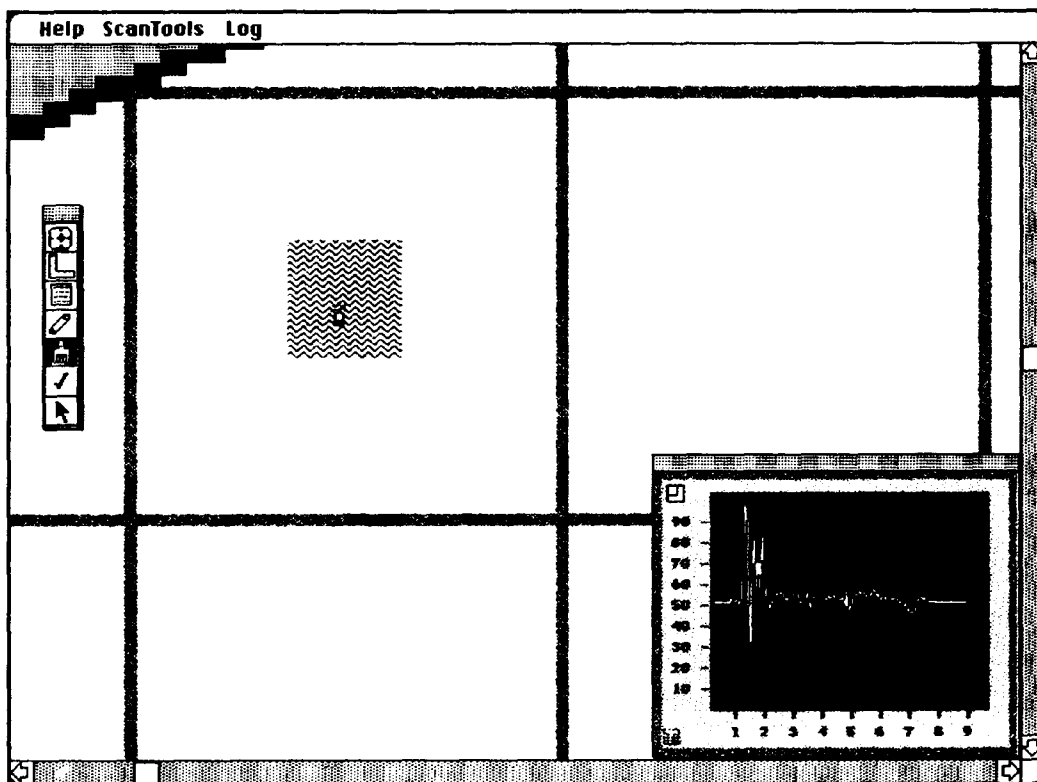


Figure 9. Inspection Display with Couplant Being Applied

Figure 9 shows the student applying couplant to the section of the radome that will be scanned by selecting the couplant tool from the tool palette, aiming the spray can, and holding down the mouse button to spray on the couplant. If couplant is not applied, the student will not get the proper BSE reading, and the simulated transducer (the cursor) will jitter to indicate the rough interface between the composite and the transducer. In this case, the couplant is water, as indicated by the spray can (as opposed to a brush). The couplant selection (water or gel) and couplant parameters, such as evaporation rate, are all parameters that can be configured by a designer and instructors. The student will typically fill in the radome section he is scanning with couplant. When this is completed, the student can select the ultrasonic transducer from the tool palette and begin scanning. As the student scans, the couplant will evaporate at a rate set by the instructor.

Several help features have been built into the Trainer to assist students that are just starting to learn this application. Figure 10 shows an on-line help agent, named Alfred, that is available to describe any of the calibration and inspection tools.

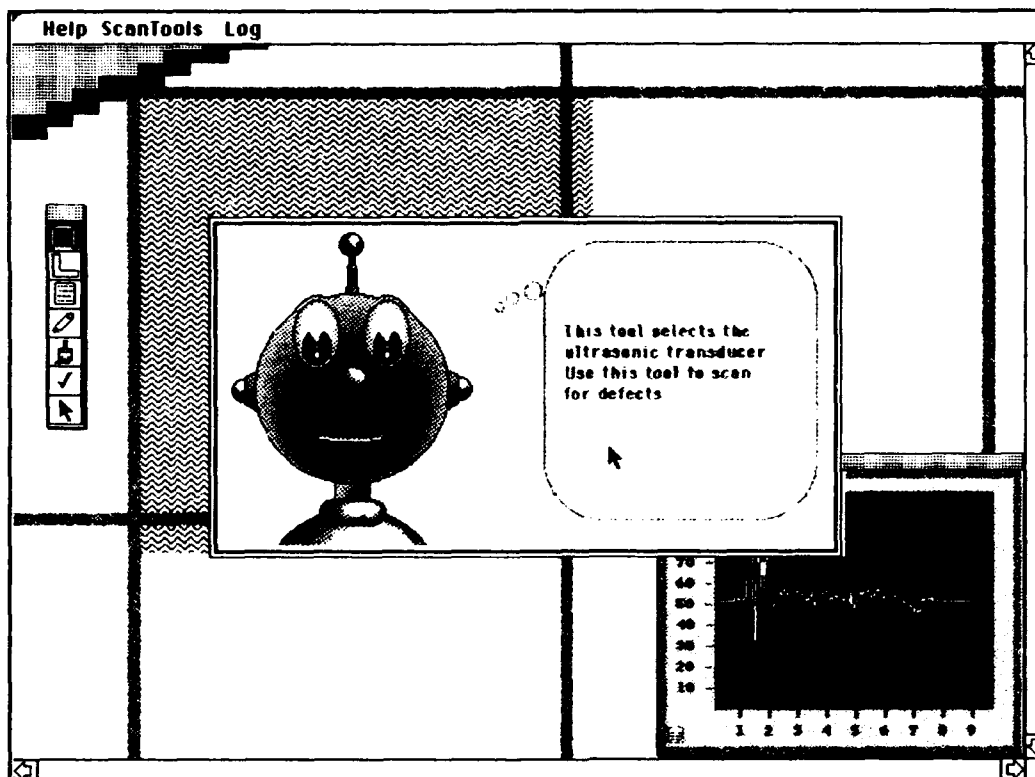


Figure 10. On-Line Help Agent Assists Student

The student uses the *Help* key to call up Alfred. To receive a description of any of the tools, the student simply selects the tool of interest and clicks on the *Help* key. Alfred will then provide a brief description of the operation of the highlighted tool. To hide Alfred, the student just clicks anywhere in Alfred's text bubble. This experimental method of providing help has been well received by students. It combines text, voice, and a visual communication agent, as opposed to a text-only presentation that is often used in on-line help systems. We expect to see this mechanism used by other programs in the future. A commercial tool has been used to develop this help function (see paragraph 4.3).

Figure 11 shows how a digitized video tutorial illustrates proper scanning procedures. As described earlier, digitized video has also been used in the calibration display. In this case, a short video segment illustrates how to scan the radome using the ultrasonic transducer. Because the tutorial is represented as real-time, digitized video frames, there is no need for a VCR. This allows the Trainer to run on any Macintosh that meets the minimum memory requirements. System operation and installation requirements are outlined in paragraph 3.3.

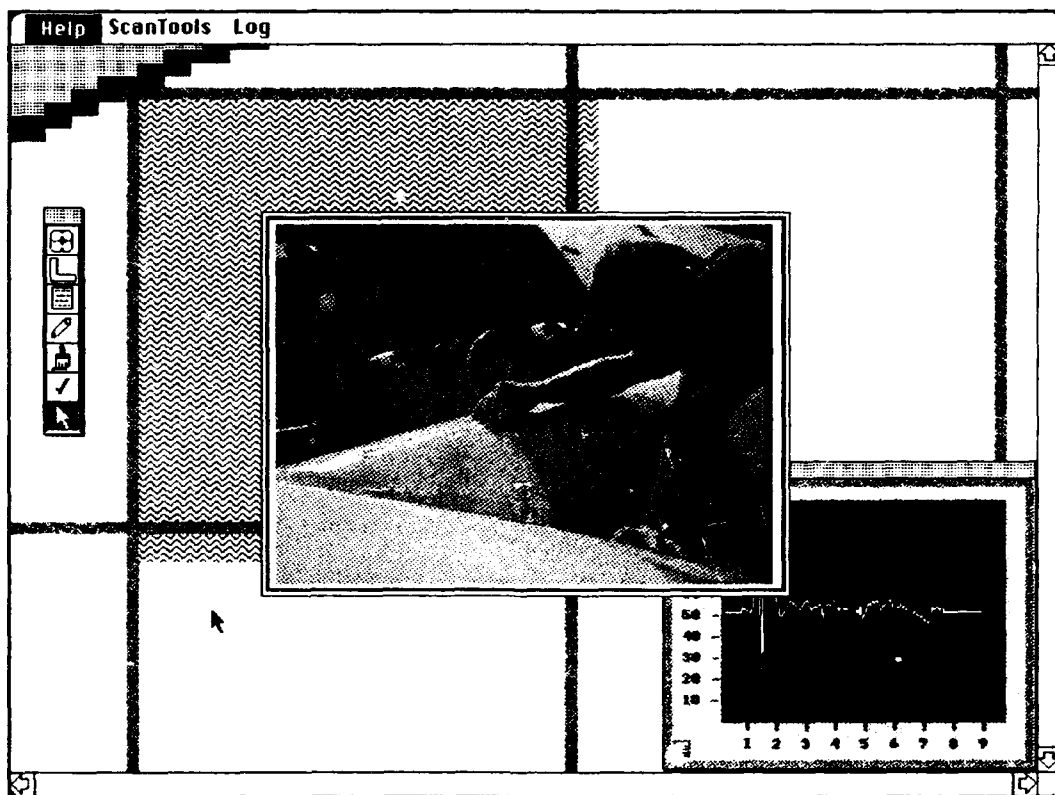


Figure 11. Digitized Video Tutorial Illustrates Proper Scanning Techniques

Returning to the inspection, figure 12 shows that couplant has been completely applied to the section being scanned. In addition, the student has selected the ultrasonic transducer from the tool palette and is scanning the selected section.

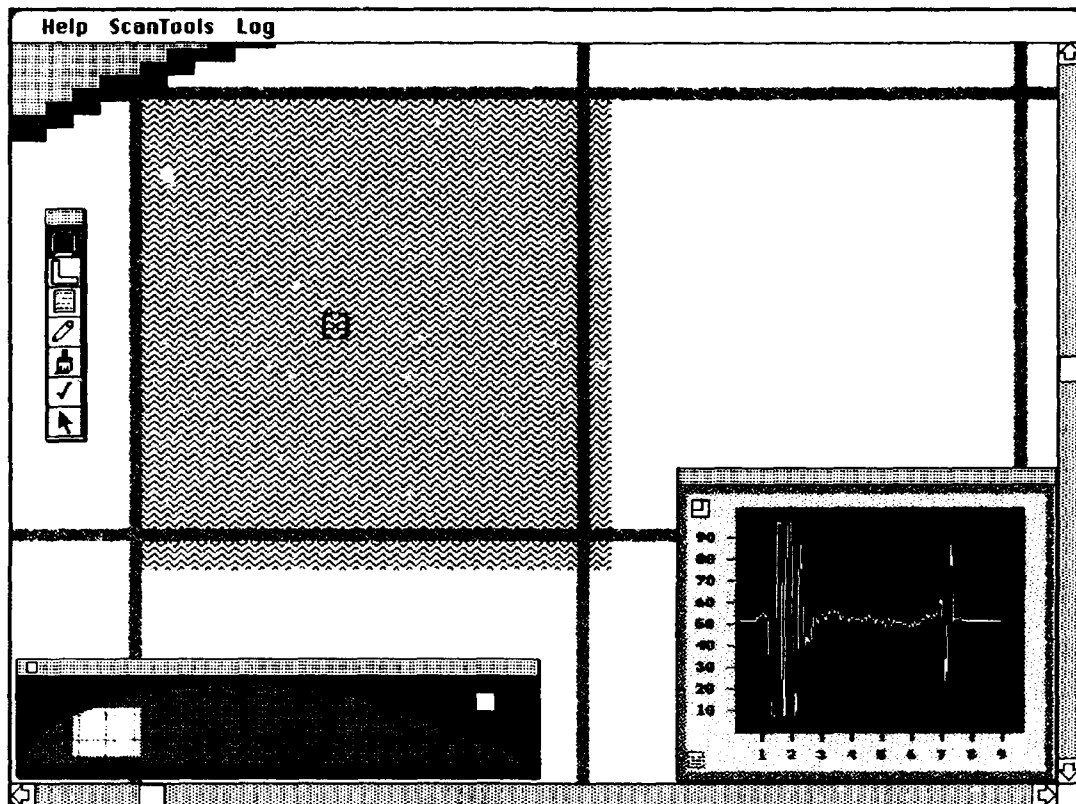


Figure 12. Student Inspects Section of Radome

As with the actual inspections, scanning in the Trainer takes time. The student must scan precisely so that no surface area is missed. As we will see later, an instructor can create flaws of any size. If the student scans too quickly, a very small flaw that only covers a few pixels could be missed. In figure 12, the signal in the ultrasonic meter is a good signal, indicating that the material has no flaws where the transducer is located. The student will scan this section, top-down and left-to-right, until he has covered it completely. While scanning, he studies the simulated meter for BSE loss and any signal changes that are indicative of a flaw. The Trainer is also equipped with a pen-mouse. The student can use the pen-mouse during scanning; it matches the look-and-feel of an ultrasonic probe more closely than the standard system mouse.

Figure 12 also shows that the student has requested a new window that provides a map of the radome (lower-left corner of display). This window serves a couple of different purposes. First, the white square area shows where the student is currently located on the radome. When zoomed in on the radome it is easy to lose perspective of where one is scanning; this window provides the perspective. The second purpose of this window is to have a way to show sections that have been inspected. When the student has completed scanning a section of the radome, he simply clicks on the section and a check mark will be recorded on the radome map.

In Figure 13, the student has located a flaw area and is in the process of specifying the boundary with the marking tool. Notice that the signal in the ultrasonic meter is different than the normal signal, as shown in figure 12. The entire radome and any flawed areas are coupled into a database of real ultrasonic signals. This is important because it allows the student to practice on real data. In terms of the Trainer implementation, flaws on the radome are objects in an object-oriented environment. When the student's simulated transducer enters the boundary of a flawed object, this generates a message that is trapped by the software so that a new signal can be generated. As in real inspections, flaws may be relatively large and easy to find, or flaws may only be a few pixels large and easily missed. The student uses the marking tool by selecting it from the palette and clicking on points where there is a transition from the normal signal to the flaw signal. The student may remove any existing mark by clicking on it. He can also use the *Delete* key to remove all the marks associated with a flaw.

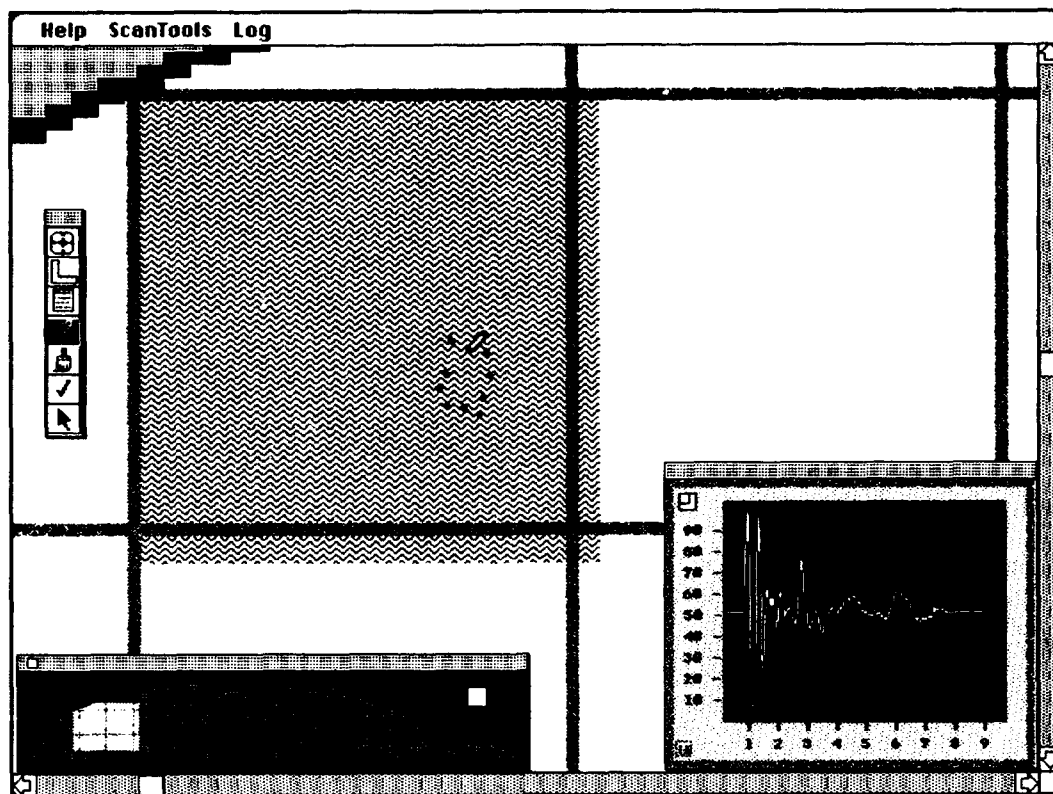


Figure 13. Grease Pencil Selected after Flaw Has Been Located

Once the student has dropped a sufficient number of marks to specify the boundary of the flaw, he fills in the flaw area. This is shown in figure 14 as the yellow area on the radome. The flaw is filled by selecting the marking tool and then manually connecting the marks or having a software algorithm draw a polygon from the marks. Typically, the automatic algorithm is sufficient. However, in some cases where the flawed shape is highly unusual, such as a figure eight, the flaw must be manually drawn.

Figure 14 also shows that the student has selected a set of rulers so that the flaw position and size can be determined and logged. The rulers are selected from the tool palette and are moved by clicking on the area of the radome where the rulers should be located.

When a flaw has been completely marked, the area is filled in and the student must log in information associated with the flaw into an inspection notebook. The student first identifies the flaw type (based on the signal) from a list of possible flaws. The list of flaws in figure 14 is not the final list and simply represents the type of flaws that a student will need to learn. (Note: A designer utility will allow the final list of flaws to be created based on actual

inspection and analysis data. The utility allows this list to be generated without any software changes. Inspectors from SM-ALC will create this list.) The student also enters flaw size and location information and any applicable comments. Flaw size and location are very important, because some flaws may be small and not warrant repair. In these cases, the flaws must be tracked carefully for any gradual changes from inspection to inspection.

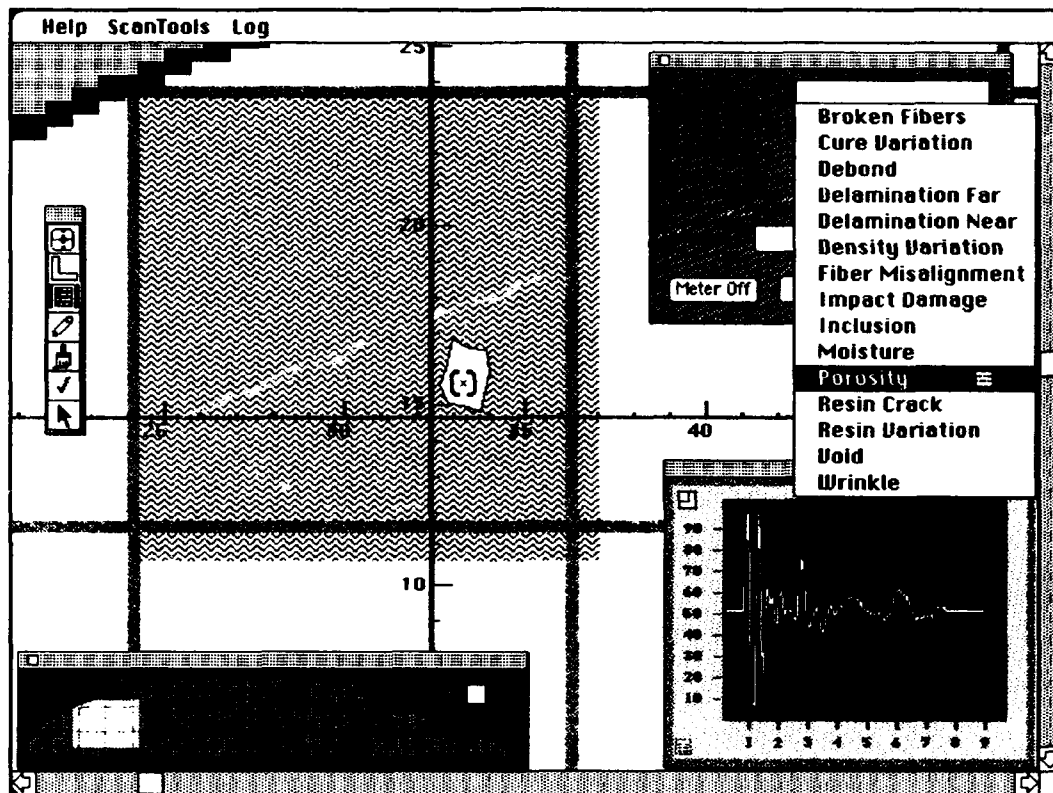


Figure 14. Flaw Outlined and Result Entered in Inspection Notebook

For students that are just beginning to learn the Milstar radome inspection, a tutorial illustrates the various types of ultrasonic signals a student must learn to recognize. A sample SEM image tutorial is presenting information on specific flaw types (in this case a porosity) to the student (figure 15). In this display, an interactive SEM image, graphic of the signal and text allow the student to visually explore the various types of flaws that may be encountered in pulse-echo ultrasonic inspection. In particular, the model signal has several 'hot-spots.' As the cursor passes over the signal, the corresponding area in the material is highlighted and a brief text description is presented. This tutorial helps solidify the student's understanding of how signals are generated from the Kevlar/polyester composite. The student can zoom in on the SEM image (figure 16).

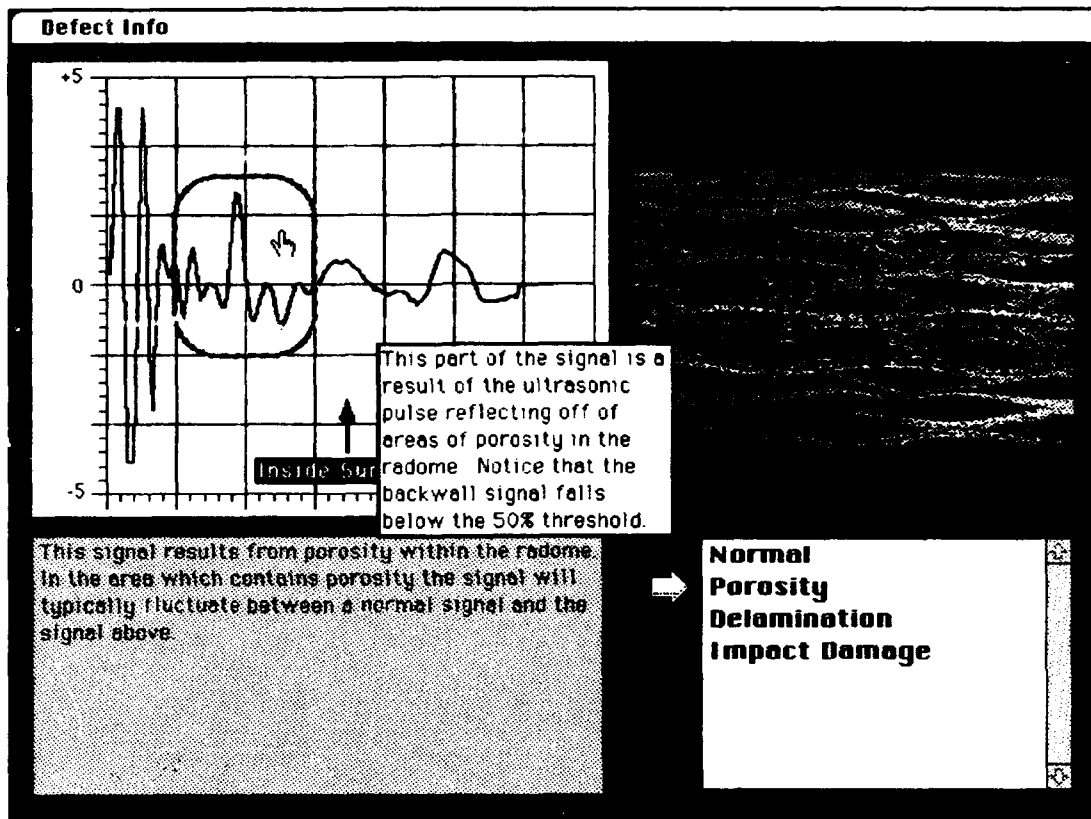


Figure 15. Signal and Image Tutorial To Illustrate Signal Types

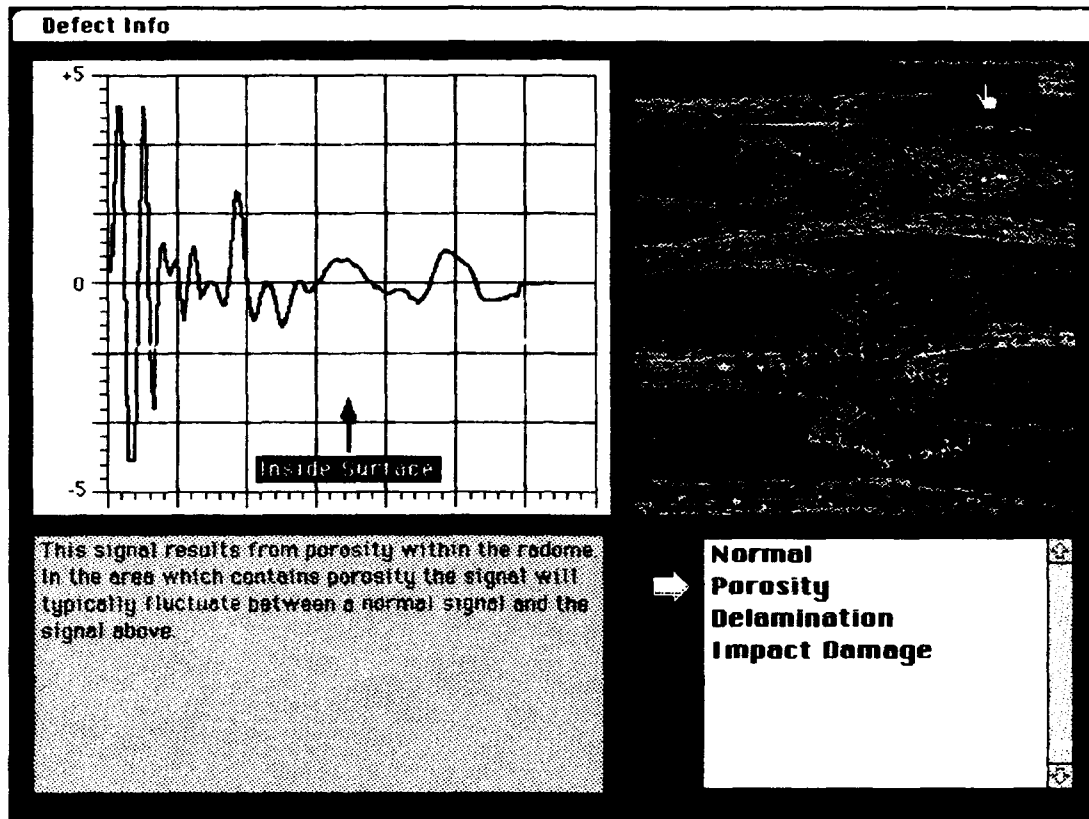


Figure 16. Microscope Image Magnified in Tutorial

3.2.2 Radome Editing

Instructors log in to the Trainer in the same way as the students, by entering an account name and providing a password. After the login sequence has been completed, the instructor is greeted with the main menu bar, as shown in figure 4. Since the account is an instructor account, the *Instructor* menu is enabled. One option instructors have is the ability to call for a list of radomes in the system. An instructor may then select a specific radome and examine the flaw areas that are associated with that radome. For example, figure 16 shows that the instructor has selected a radome and has clicked on a small flaw at the front of the radome for more information. The instructor can recall either the flaw name or a window that provides all the information about that flaw, including the ultrasonic signal. Flaws are selected by clicking on the area of interest. The flaw window is shown in figure 17. This function provides a convenient way for an instructor to browse through radomes in the database.

Editor Help Instructor Inspection

DEFECT INFORMATION

Radome Name:

Defect Name:

Defect Type:

Skill Level:

Defect Height:

Defect Width:

Inches From Front of Radome:

Inches From Bottom of Radome:

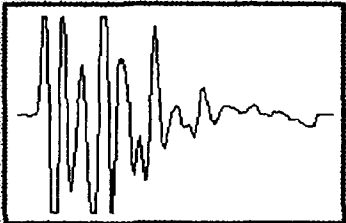
Model Ascari: 

Figure 17. Instructor Examines Flaw Areas in Radome

Instructors also have the capability to examine student results, either during or after an inspection. Figure 18 shows that an instructor is comparing the inspection results entered by a student to the flaw information that is actually associated with the radome. In particular, the instructor is interested in how well the student evaluated the signal type, recorded the location and size of the flaw, and outlined the boundary of the flaw. In this case, the student completed only one section of the radome, as shown by the single check mark in the radome map. If the student had finished the inspection, the radome map would show check marks in all squares. For a student that is just starting to learn the inspection, we anticipate that the instructor will monitor his progress closely. Notice that the map does not show the flawed areas, so the student can see only the area of the radome that has been zoomed-in.

Note the similarity between figures 18 and 14. These displays are identical except for the instructor's capability to show the real flaws. Consequently, an instructor can log in and use all the tools in the palette to work with a student. In fact, there are many ways that this display could be used, depending on the training procedures that are defined for this inspection. For example, new students could be given a radome to inspect with the flaws shown. This would

provide practice using the tools and getting familiar with the signals. Alternatively, an instructor could show a student the real flaw to indicate where to look and then hide the flaw to make the student work at marking the flaw area. Note that student-entered and database flaws can be toggled for ease of viewing. This function can also be run by the instructor when a student is logged into the Trainer. In that case, an instructor password is required whenever the function to view flaws is desired. The function is disabled automatically when the flaw areas are hidden (no instructor logout is required).

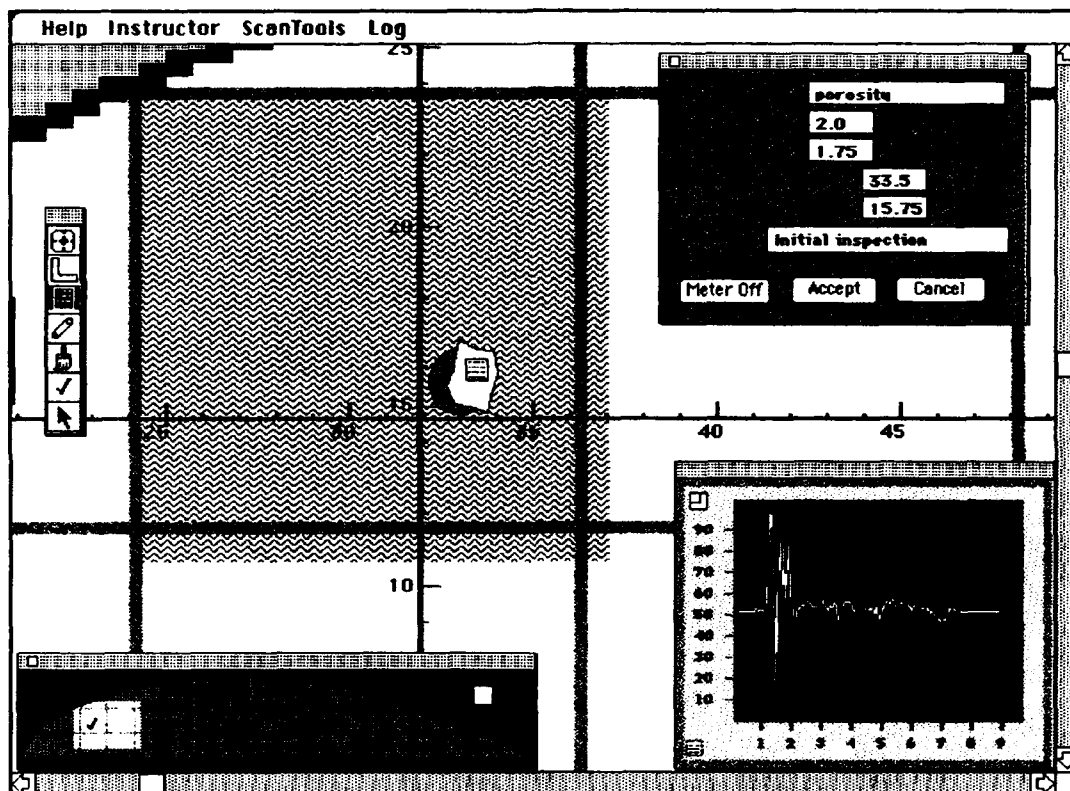


Figure 18. Instructor Reviews Student Inspection Results

One of the Trainer's most powerful features is the capability for instructors (and designers) to create radomes with arbitrary combinations of flaws. In particular, a radome can include any number of flaws of any type in the signal database, and flaws can be given any shape, size, and location that the instructor requires. The Trainer also error-checks as an instructor creates a radome to ensure that flaws are not incorrectly placed, such as on a fastener or off the radome. All the flaw and radome information is stored in a database that exists as part of the Trainer application. This allows the database to exist completely transparent to students and

instructors. Figure 19 shows the radome that was used earlier in the student user thread (paragraph 3.2.1). The background of this display (the radome graphic) is identical to the inspection display. However, the tool palette (left side of display) contains radome-editing tools for the instructor. The top tool is the pointer (or selection) tool, which is used to click on a flaw for the attribute list. The next five tools are drawing tools that allow the instructor to create flaws of different shapes. The five drawing tools are square, rounded rectangle, ellipse, polygon, and free-form. The free-form drawing tool is being used in figure 19 to draw a complex flaw shape. The last two tools in the radome-editing tool palette represent delete and drag functions. The delete function allows flaws on the radome to be removed; the drag function changes the location of a flaw. We expect that designers and users will use the radome editor to create a library of radomes that will be used for training.

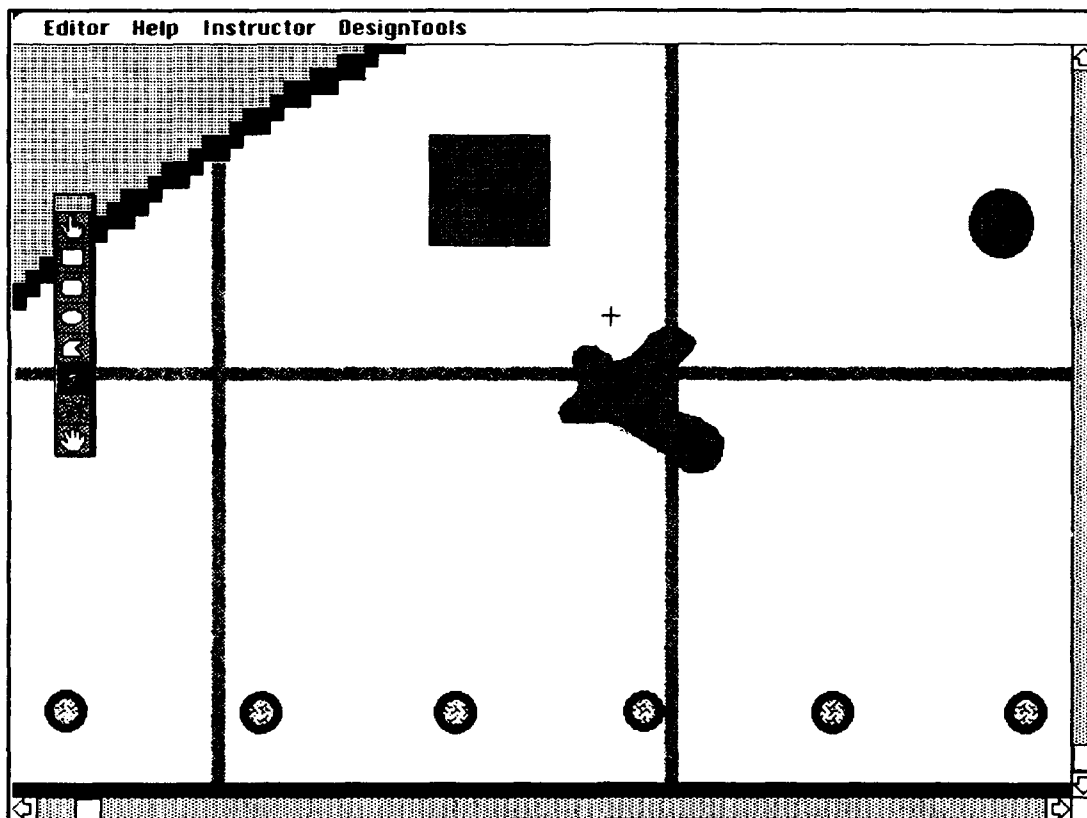


Figure 19. Instructor Edits Radome in Database

In figure 20, the instructor has completed drawing the shape for a new flaw. This causes a dialog to be activated that prompts for the flaw (signal) type. Flaw name, skill level, height,

width, and location (inches from front and bottom) are automatically derived from the flaw graphic. The *Cancel* key will cause the Trainer to close the window and undo the new flaw drawing. If the flaw had previously existed, *Cancel* simply would close the attributes window and would not save any changes. Only flaw types can be changed in the window. Location is changed using the drag function.

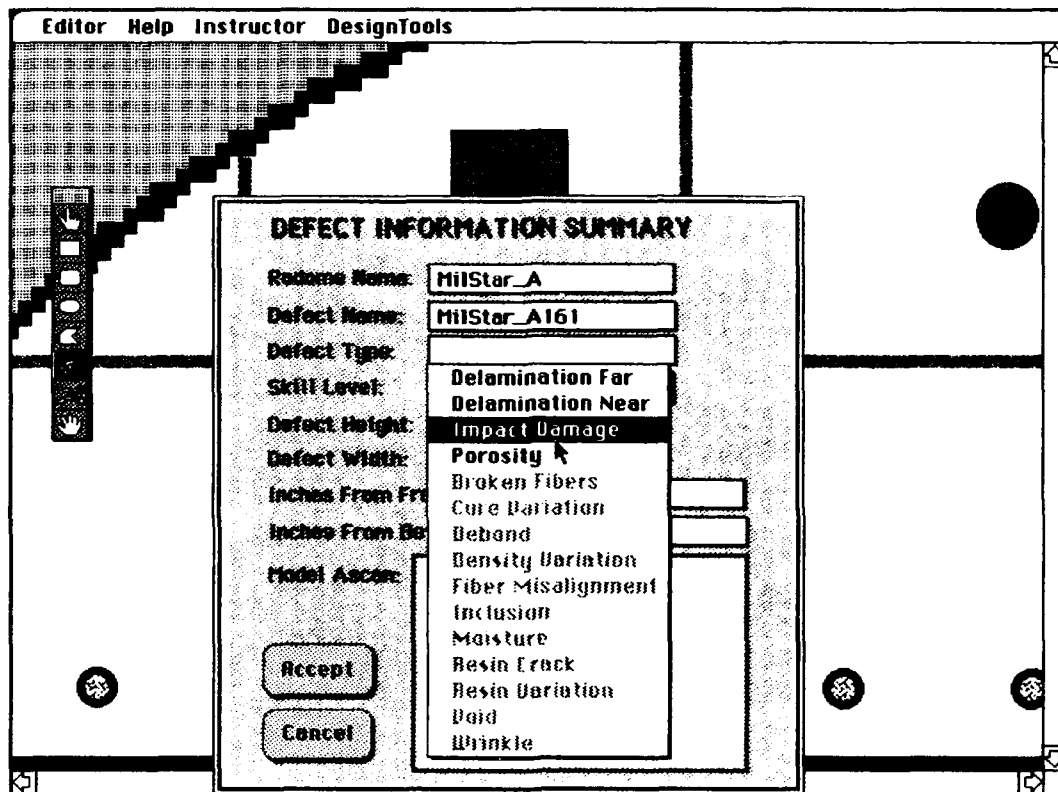


Figure 20. Instructor Selects Flaw Type To Add to Radome

Instructors have functions to maintain user accounts in the Trainer. Figure 21 displays the user accounts in the Trainer. This display and all associated menu functions are only accessible to the instructor and designer. For each user account, the name, password (if available), access level, and last date and time of login are listed. Passwords are shown for accounts that are at a lower level. That is, instructors can view the passwords of student accounts but not instructor or designer accounts. Similarly, designers can view the passwords of student and instructor accounts but not other designers. Menu functions allow user accounts to be added, removed, and modified. In addition, sort functions permit sorting by name and access level. The

training results of each student are associated with their account and can be accessed by instructors as well as designers. A print function allows the list of student accounts and associated passwords to be provided in hard-copy form.

User Sort				
Name ✓	Password	Access	Last log time	Last log date
Don Bailey	!!!!	Designer	5:03:51 PM	11/11/91
Donna Ballard	!!!!	Designer	5:08:30 PM	3/3/92
Donna DeRose	pass	Student	9:29:51 PM	3/22/92
Donna DeRose	!!!!	Designer	4:44 AM	3/22/92
Ike Spencer	pass	Student	7:17:01 PM	4/11/92
John McNamara	John	Student	8:56:40 AM	12/26/91
John Wilson	!!!!	Designer	8:55:42 AM	3/22/92
Peter Nesky	pass	Instructor	9:37 AM	5/14/92
Scott Blodget	pass	Student	11:07:21 AM	4/11/92
Steve Fortuna	Steve	Student	9:49:17 PM	4/11/92
Steve Harris	pass	Student	9:32 AM	5/14/92
Stu Jolly	pass	Instructor	11:08:25 AM	4/29/92

Figure 21. Instructor Lists User Accounts

Figure 22 shows the couplant control window that allows instructors and designers to tailor the couplant parameters from site to site. In particular, the couplant control window allows for selection of couplant type (water or gel), dissipation time, couplant shrink rate, couplant color, and simulated transducer jitter delay. The transducer jitter is the amount the cursor jitters when the student is positioned outside the couplant. When the student moves the transducer inside the couplant, the cursor moves smoothly, simulating the transducer actually gliding over the couplant. The instructor can change any of these parameters and accept or cancel the changes. An *Accept* means that the changes are permanently recorded and the Trainer now operates in accordance with the new parameters. A *Cancel* restores the couplant parameters to the values

they had prior to calling up this window. A *Cycle* function allows the instructor (or designer) the chance to see the couplant cycle from fully applied to fully dissipated. We expect that designers will provide a default set of couplant parameters. Each site will tailor the set depending on the operating conditions at the site.

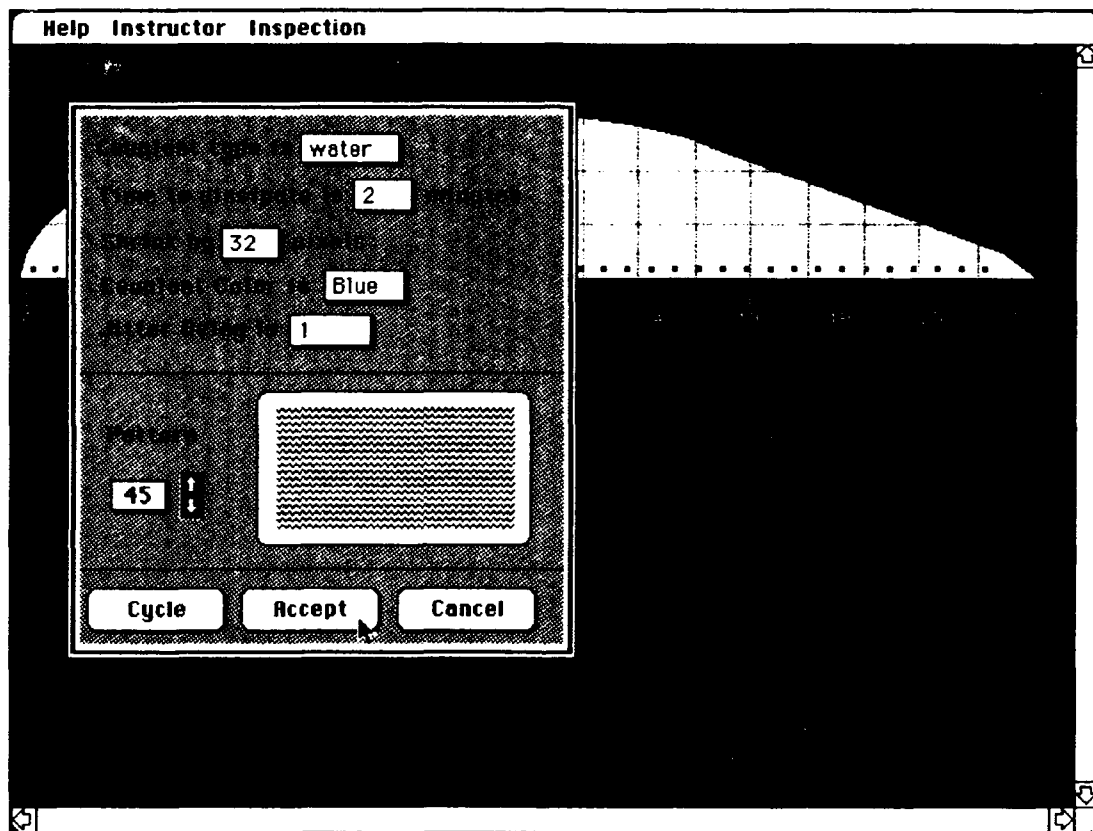


Figure 22. Instructor Reviews Couplant Parameter Settings

3.2.3 Trainer Configuration

One function is available only to designer users. Designers have the capability to import new ultrasonic signals and specify associated flaw attributes. Figure 23 shows the designer utility that accomplishes this function.

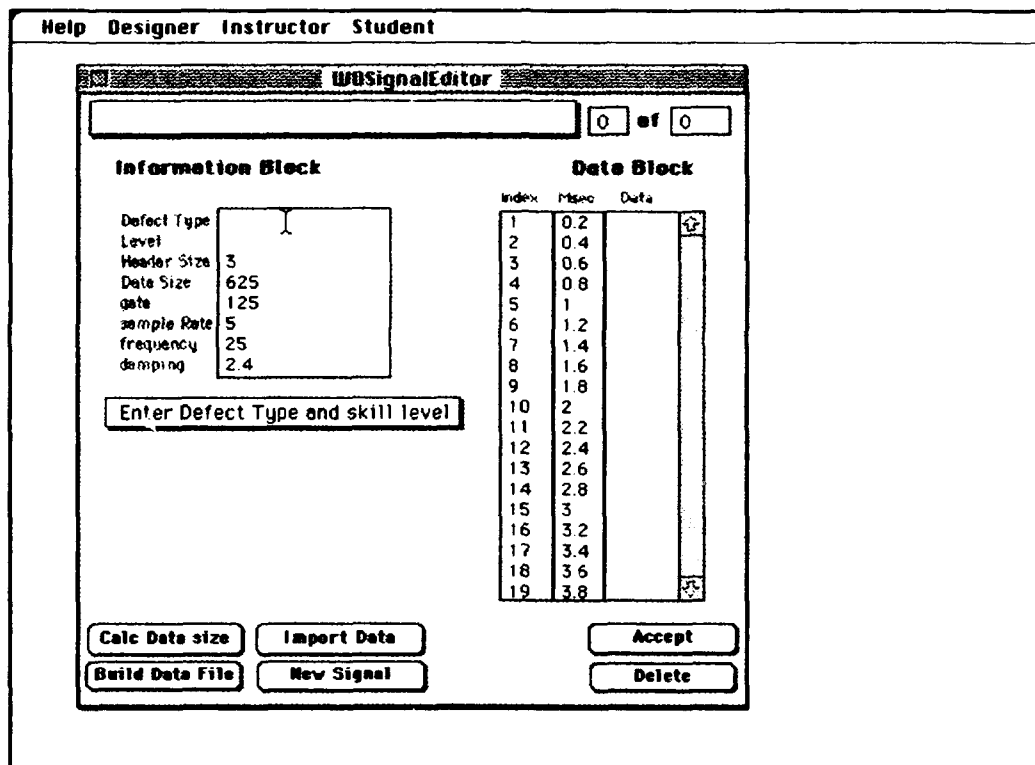


Figure 23. Designer Activates A-Scan Database Utility

The designer enters the flaw name and skill level into the *Information Block*. Then, the designer specifies the directory and file where the A-scan (amplitude over time) is located. The utility reads the digital waveform from the specified file, and the signal (data points), flaw type, and skill level are permanently recorded in the signal database. The new signal is now part of the Trainer, and instructors can create radomes for students that contain the new flaw type. Figure 24 illustrates the display after the designer has specified and read the data for a new flaw type, an inclusion.

The capability to import new ultrasonic signals is one of the most important functions of the Trainer; a considerable design effort was required to provide this capability. Because this inspection is using a new advanced composite, new signal types have been acquired as the Trainer was being prototyped. We anticipated that new signal data would become available after the Trainer prototyping was completed. Designer users, such as SM-ALC, required an easy way to change the signal and flaw database in the Trainer. This utility provides the capability and does it without requiring software changes. Designers must only provide the A-scan data file (the signal data). All associated student and instructor menus and functions are

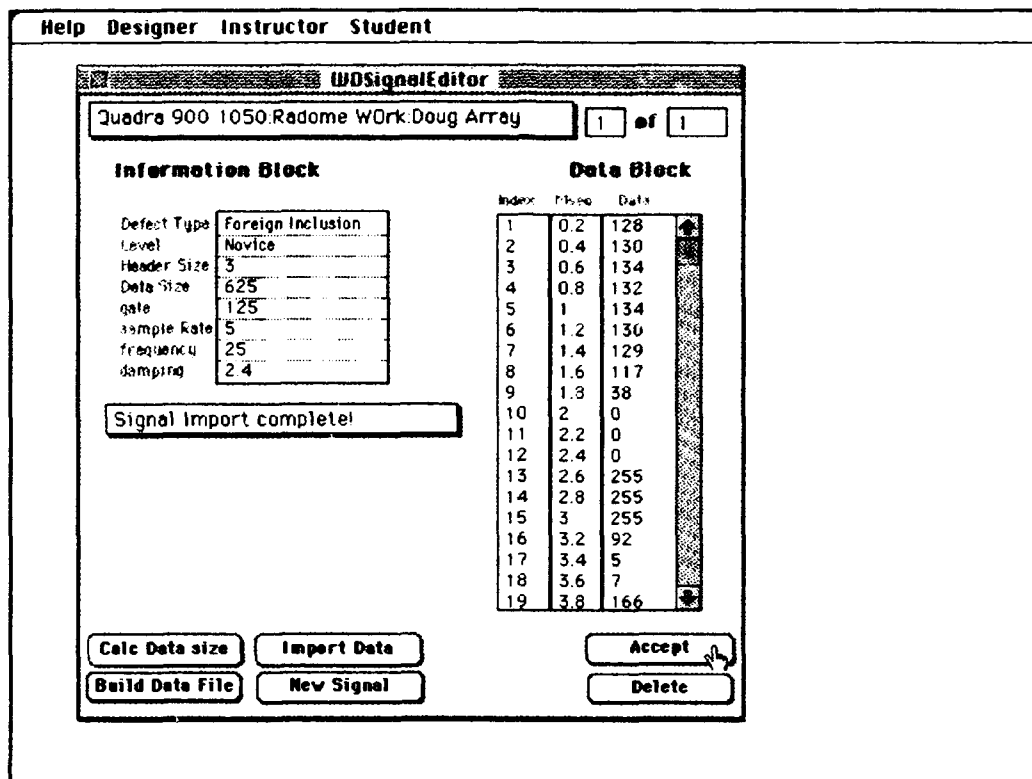


Figure 24. Designer Adds New A-Scan to Database

driven off the flaw and signal database. When the designer adds a new flaw, the Trainer is updated accordingly. Similarly, if the designer removes a specific flaw, the menus and functions are updated accordingly. Radomes with flaws that are no longer valid are flagged appropriately.

3.3 SYSTEM REQUIREMENTS AND INSTALLATION

We are developing the Trainer prototype to run on an Apple Macintosh Computer. The software uses colors to accurately simulate the real-world test equipment; therefore, a Macintosh II family of CPUs must be used. The system requires a minimum of 8 megabytes of memory, an 80-megabyte hard disk drive, and a 13-inch color monitor. The configuration is considered to be a baseline system in the Macintosh II line of computers. Although we could have used a low-end Macintosh (such as a cx or si), we strongly recommend a Macintosh fx or Quadra series computer. The Trainer uses advanced multimedia techniques and real-time signal processing, so faster CPU performance is highly desirable. The Trainer had originally been

developed and tested under Apple's 6.07 operating system; we are currently testing under System 7. Our testing should be completed shortly after this document has been published.

The Trainer prototype software will be supplied on three 1.2-megabyte floppy diskettes. Disk one will contain the Trainer program; disks 2 and 3 will contain digitized tutorial movies. To install the Trainer on the Macintosh, the user will create a new folder named "Radome Trainer" and copy all diskettes to the new folder. After all the files have been installed, the user should store the original disks in a safe area. Upon completing the installation process, the user can begin a new training session by clicking on the "Trainer Program" application. A forthcoming user's manual will detail installation steps.

SECTION 4

DEVELOPMENT APPROACH

4.1 EVOLUTIONARY DEVELOPMENT PLAN

The Trainer is an experimental application that has been prototyped for the Milstar Program. The result of this work will be a training system that can be used by SAC field inspectors to train for the Milstar radome inspection. It is instructive to compare the initial objectives of this effort and the resulting prototype system. Initially, this work was started because of the lack of a training reference for the ultrasonic signals in this application. We wanted to develop a library of ultrasonic signals and a visual, graphic presentation for presenting this information to student inspectors. Other possible uses for a Trainer were sketchy at best, and could only be refined after meeting with inspectors from SM-ALC.

The Trainer has been developed (prototyped) using a technique called evolutionary rapid prototyping (described below). This technique has allowed MITRE and SM-ALC to prototype and evaluate key functions early (ultrasonic signal library) and then refine other requirements and evolve a design that capitalizes on existing multimedia technology. Functions, such as digitized video tutorials, scanning and calibration process simulation, and the radome editor were not identified at first. Rather, they were defined and evaluated incrementally during the prototyping.

The Trainer has been an ideal application for rapid prototyping. First, the multimedia technology that has been applied is relatively new; most products have been introduced during the development or have been on the market for only a short time. Second, the Trainer is taking a different approach to NDI training by providing hands-on practice using simulated test parts and test equipment. Most training organizations use real parts and real equipment, and there are limitations in that approach (as described in sections 2 and 5). Third, the Milstar inspection application is very new. Inspection data is limited and procedures are evolving as more and more inspections are performed. As SM-ALC inspectors discover attributes about the radome, the Trainer has been adapted as required.

Traditionally, in systems development, requirements are derived through various fact-finding methods. A systems analyst attempts to learn exactly what takes place in the current system, determines and fully documents what should take place, and then provides recommendations to the customer, users, and management. Conventional fact-finding techniques include interviews, observation, data collection, questionnaires, timing studies, and research, to name a few. Unfortunately, conventional fact-finding techniques do not adequately emphasize existing and emerging technology that, if applied, may solve a user's problem in a completely novel and unexpected way.

Recent advances in commercial hardware and software technology provide affordable capabilities that only a few years ago were not available to most users. Today's high-performance microcomputer technology can integrate images, voice, video, and text within graphical displays. Because growth in computer technology occurs so quickly, users are often unable to stay current and may not know what is available. Vendors are highly motivated to

advance their products as quickly as possible to gain technical and marketing advantages over competitors. Product life cycles are very short, so in order to remain competitive, vendors must typically produce major upgrades every 6 to 9 months and introduce new products every 12 to 18 months. The multimedia market is particularly competitive. If exploited properly, this creates fantastic opportunities for prototyping new applications.

Users and developers can refine requirements together, based upon products that are commercially available. Typically, users and developers must iteratively explore the problem domain to gain a sufficient understanding of the requirements, a situation which almost never exists at the beginning. As mentioned earlier, a number of fact-finding methods can be applied. The user must be introduced to applicable commercial products that will demonstrate the capabilities that address the user's key operational requirements. When a commercial product meets all or part of the requirements, development effort, risk, and costs (since the product already exists) can be saved. Commercial software products will also typically drive out design solutions and will have particular user interfaces and functional capabilities. The developer must carefully evaluate each product to assure that it properly addresses some subset of the user's requirements, provides an acceptable design solution, integrates well with other products, and is sufficiently flexible for future system growth. In this light, the developer's job is to select commercial products that best address the user's needs and integrate the products to provide a system solution for the user. This is in contrast to the developer writing a lot of custom software, an activity that is associated with conventional development, and is time-consuming and risky.

There is an important message here. In developing new systems, requirements and design uncertainty always exist. The user often understands key operational requirements but cannot articulate the detailed technical requirements needed to build a system. Developers and users should see what currently exists, commercially, that can be leveraged against the problem. The fit may not be perfect, but if a mix of commercial products can be successfully integrated to provide a 90-percent solution, the user can realize enormous savings in development costs, risk, and schedule. Conventional fact-finding methods are still necessary; however, the focus is different. A technology-oriented approach to fact-finding will drive detailed requirements in a direction that existing products can best satisfy.

Figure 25 illustrates the process of accelerated systems development through rapid prototyping. The diagram and process model are based on the spiral model for software development created by Barry Boehm in the early 1980s [2]. The diagram should be read from the center, spiraling out, potentially leading to an operational system. There are a few general points we can begin with in discussing this model. First, note that the iteration is carried through the spiral and that development process structure is introduced as a project works its way along. Early on, the goal of the prototyping should be to establish the key requirements and technical feasibility. The purpose, scope, and high-risk areas of the development need to be defined and clearly articulated to the user as quickly as possible. Inherent in the process is communication that revolves around the prototyping and encourages the user, customer, and developers to work closely together. This is useful in managing user's expectations, since they become educated in the process. Another important point about the model is that by building incrementally, multiple points exist in the process where technology can be injected. Also, multiple prototype versions allow for spin-off of quick display prototypes or design prototypes that focus on particular risk areas. For example, if a display prototype is the scope of the effort, only the

inner part of the life cycle is applied. Incremental delivery of capability to users is desirable because it may satisfy immediate needs, and it also lets the user refine requirements based on hands-on experience.

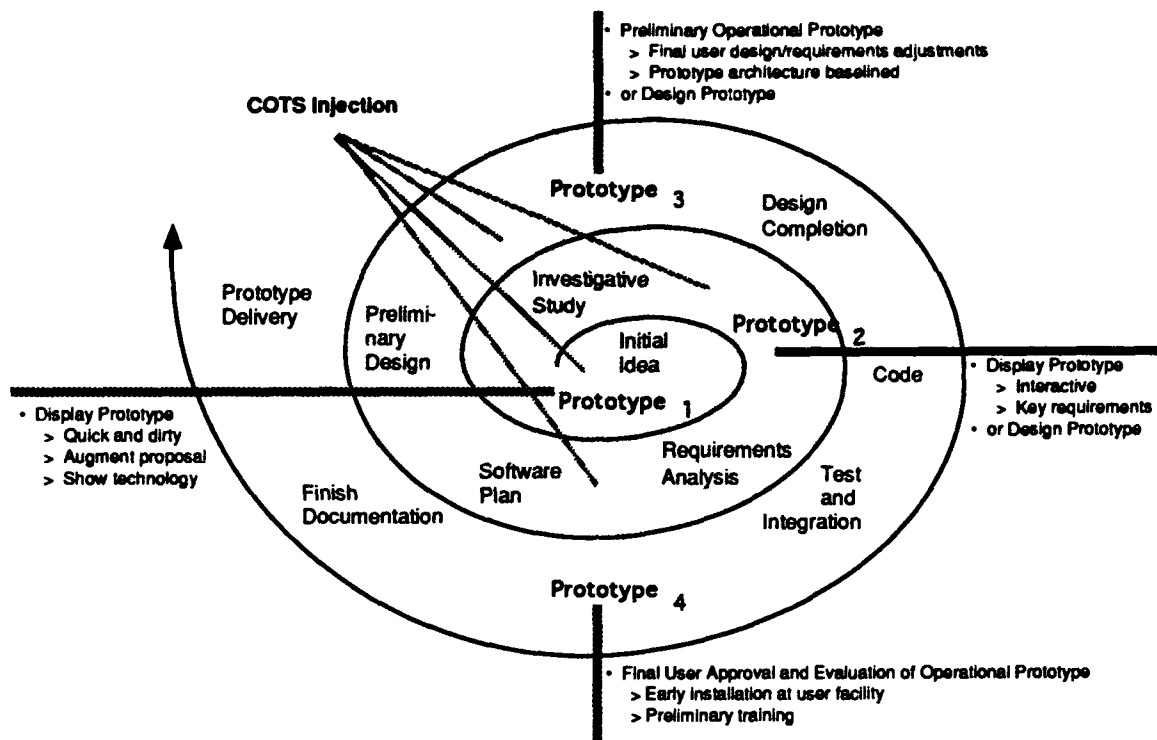


Figure 25. Evolutionary Rapid Prototyping

As the prototyping progresses and the requirements begin to stabilize, developers must create an architectural design that will allow for future growth and change. Prototype architecture is particularly important in rapid prototyping efforts that capitalize on commercial products. The reason, quite simply, is that most commercial products cannot (and should not) be changed by system developers. Ideally, a prototype architecture should be based on readily separable components. Interfaces between components must make little or no assumptions about individual packages. That is, a detailed understanding of a package should not be required by other areas of the system. Additions and enhancements to the system should be more commercial products, and again, some modicum of custom software. While these general, object-oriented guidelines are noble, deriving a versatile architecture can be a challenging task. Planning must acknowledge that prototype architecture is critical and make sufficient allowances in the schedule so it can be worked carefully.

The prototype Trainer will be used to help train field inspectors on the Milstar radome inspection. However, the Trainer is only part of a formal training program that is being planned. SM-ALC will provide a review of pulse-echo ultrasonics, explain the attributes of the Kevlar/polyester composite, and help the student acquire enough knowledge to start using the Trainer. Transition of the Trainer will likely be carried out with SM-ALC providing support for the prototype once MITRE is finished. SM-ALC already has a development system with the software installed. We anticipate that most, perhaps all, changes to the Trainer can be accomplished by manipulating the database. For example, new ultrasonic signals can be added with changing software. Nevertheless, SM-ALC will be trained to make small changes to the software, if needed.

4.2 ARCHITECTURAL DESIGN

In this section, we present a high-level description of the architecture design and focus on three specific areas of the design: function/data interaction, commercial off-the-shelf (COTS) tool interfaces, and database structures. Figure 26 shows a high-level architecture developed in the early stages of the prototype. The shaded cloud-like graphics represent the functional four main areas; the oval graphics represent specific functions within each main area. The rectangular objects in the center of the figure represent data stores for the system, and the arrows represent the paths in which data flows between the functions and data stores.

During the design of the system, we placed much attention on isolating the data from the runtime code, because NDI scientists were collecting and analyzing the data in parallel to our efforts. By creating a flexible architecture that could foresee major changes in data, the Trainer can easily adapt to new discoveries made in the laboratory. An example of the data isolation can be found in the "DESIGNER UTILITIES" area of figure 26. The function is provided to import A-scan data that had been collected in the field. Functions were developed to read ASCII files from a disk that contained digitized A-scan amplitude waveforms. A conversion function converts the amplitude points into the Trainer's internal format and stores them in the "Ultrasonic Signals" data store. The data is then used by other areas of the software for simulation and analysis purposes. Because functions make no assumptions on characteristics of the data, adding and modifying "Ultrasonic Signals" has been simplified.

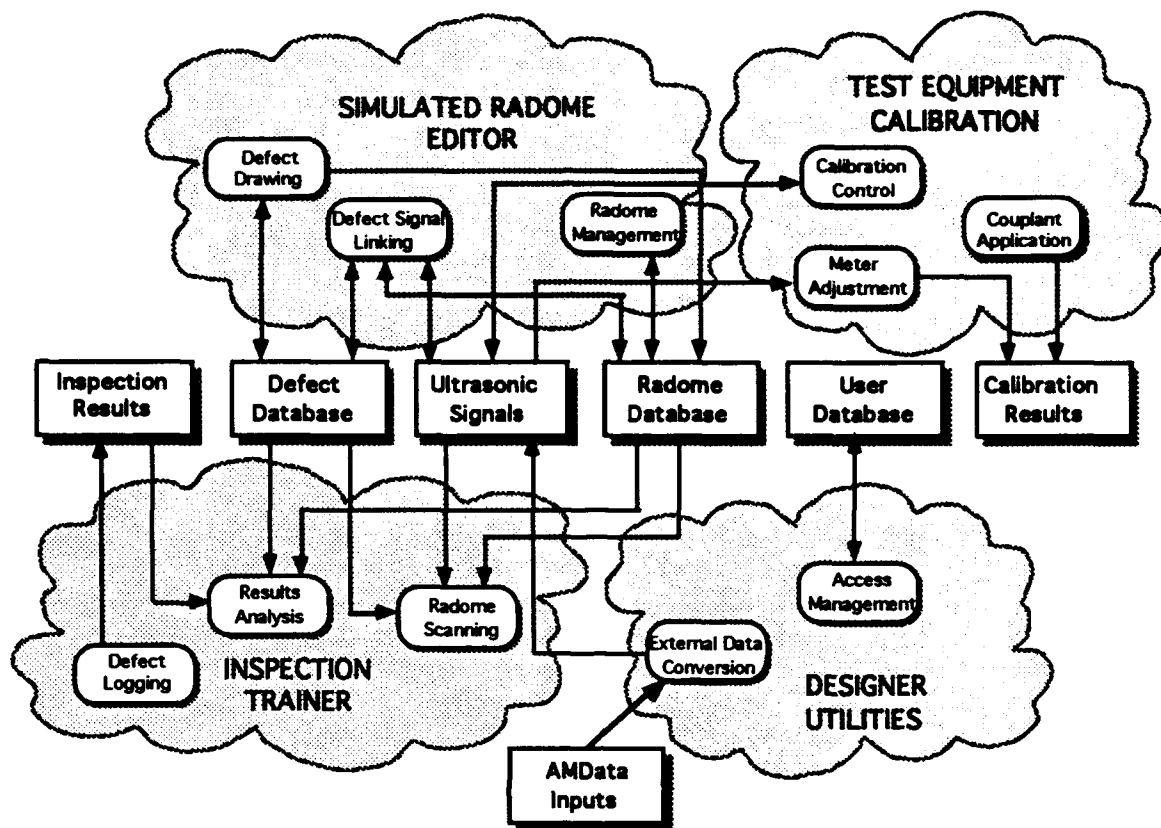


Figure 26. System Architecture

The Trainer prototype makes extensive use of COTS tools and compiled code routines in its implementation. SuperCard was designed with an open architecture and provides a gateway for invoking add-in applications. This is accomplished by using interfaces called Xcmds (eXternal CoMmanDS) which are typically provided with the COTS products. The Xcmds are imported into SuperCard as resources and can be invoked as if they were internal SuperCard commands. Once invoked, Xcmds supply the glue between SuperCard and the COTS applications and are also used to create time-critical code. Xcmds can be developed in 'C' or with custom Xcmd development tools to produce fast compiled machine code. The Xcmds can then be called from SuperCard to perform speedy processing.

Xcmds is the standard mechanism used to interface COTS packages. Figure 27 shows the interfaces between SuperCard and various COTS tools used to develop the Trainer. The Xcmd COTS interfaces are called from SuperCard with the appropriate parameters. Once invoked, the Xcmd interface takes over and activates any system drivers and opens program resources. In figure 27, SuperCard is linked to the InterFACE (Inter Facial Animation Construction Environment) Xcmd. InterFACE is used in the Trainer prototype to provide talking on-line

help agents. SuperCard calls the InterFACE Xcmd by passing agent information and speech phrases as parameters. The InterFACE Xcmd now has control of the system and handles the processing needed to animate facial expressions which produce synthesized voice. To accomplish this, the InterFACE Xcmd makes calls to the RAVE (Bright Star Technology) and MacInTalk system drivers, which gather agent resources and initiate agent communication through a SuperCard window.

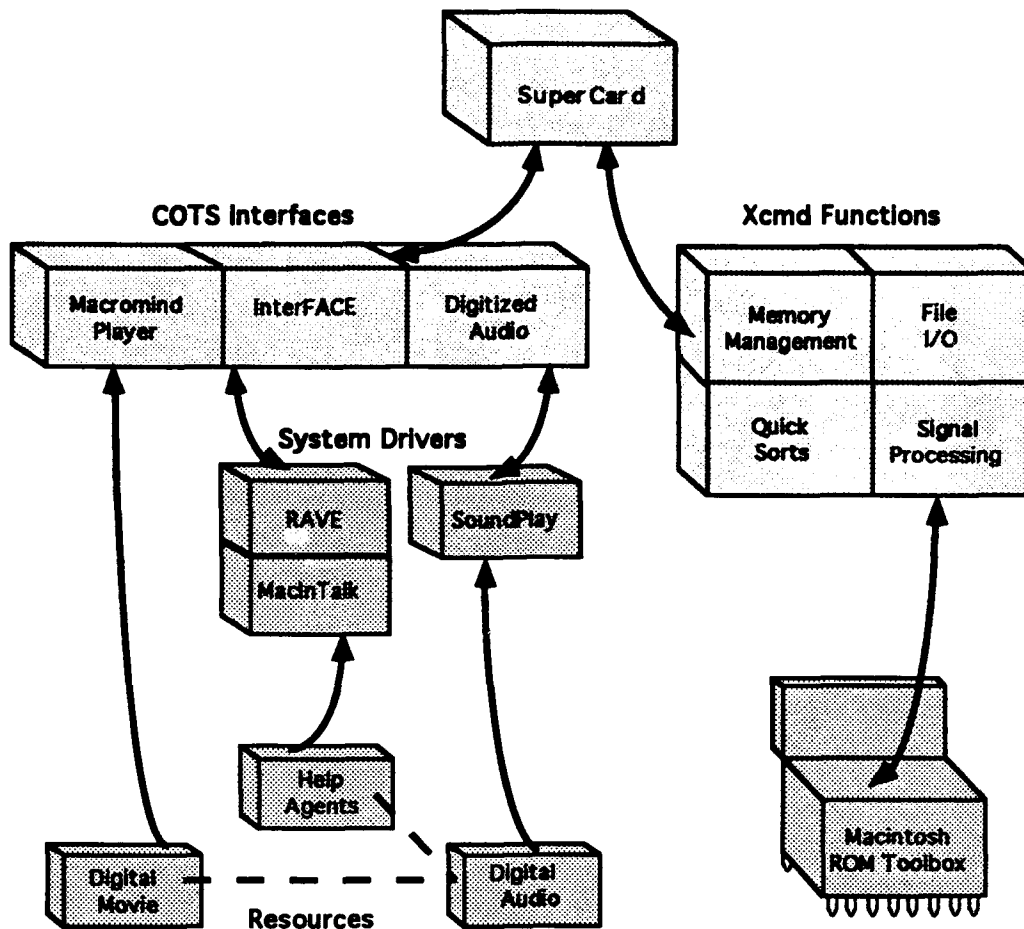


Figure 27. COTS Interfaces

Xcmds are also used to create compiled code for time-critical computing. In figure 27, SuperCard has the capability of calling developer-defined Xcmd functions. SuperCard can pass up to 16 parameters to the Xcmd code resource, and the Xcmd has the ability to access

SuperCard structures such as cards and fields. This is accomplished with mechanisms called binary and text callbacks. When the Xcmd needs the service of a SuperCard, a callback is initiated by the Xcmd asking SuperCard to perform a function. For example, if the Xcmd wanted to change the position of a button in a SuperCard window, the Xcmd would simply instruct SuperCard to perform the task rather than perform the cumbersome task itself. Unlike SuperCard, the Xcmd has direct access to the Macintosh ROM Toolbox. This is useful for performing sophisticated file I/O and memory management that is not inherent in SuperCard.

Figure 28 shows the structure of the three main databases for storing simulated radomes, defects, and ultrasonic signals. The *Radome Database* contains information specific to simulated radomes. When the instructor creates a new simulated radome, a new radome record is also created and initialized with its creation time and date. When a defect is drawn, a new defect record is created and stored in the defect database establishing a relation between the radome record and each new defect. The software also determines the skill level of the defect and places it in the appropriate field in the *Defect Database*. The skill level field values are then used to assign a skill level to the radome.

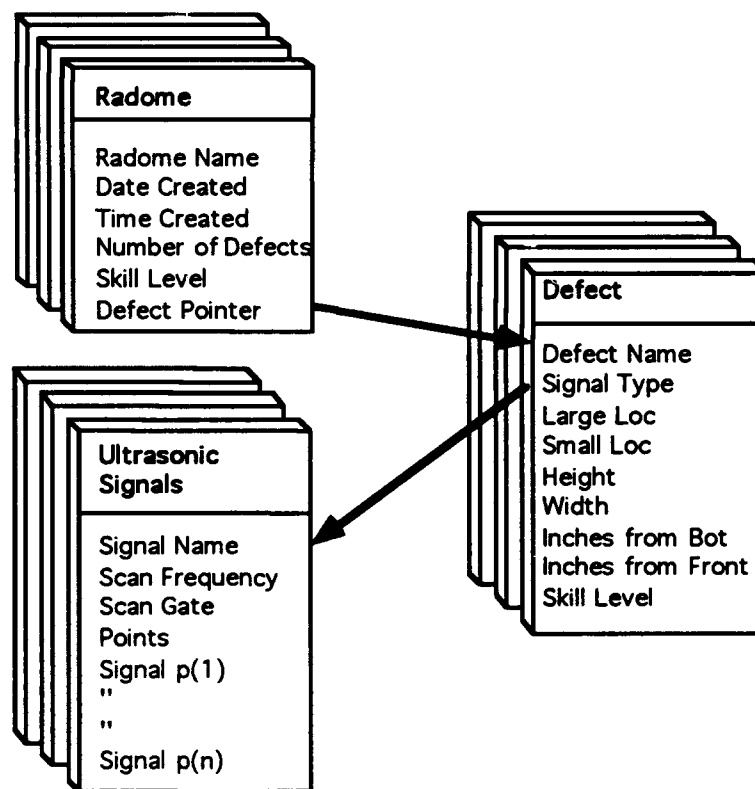


Figure 28. Database Structure

Each defect record in the database relates to an ultrasonic signal record in the *Ultrasonic Signals Database*, where information is used to recreate the ultrasonic waveform in the simulated meter. When new data is collected in the field, it is converted and added to the database. In a database-driven design for radome and defect information, simulated radomes can be transported via diskette to numerous NDI shops.

4.3 DEVELOPMENT TOOLS AND UTILITIES

In this section, we present a small subset of tools used to create the Trainer prototype. In prototyping the Trainer, we used COTS software development tools extensively. The selected COTS tools, representing the latest technology in multimedia and hypertext environments, were used in different ways, but they all interfaced to SuperCard. Because we used so many tools in this project, we will only discuss some of them. Figure 29 graphically depicts the tools we used to develop the Trainer.

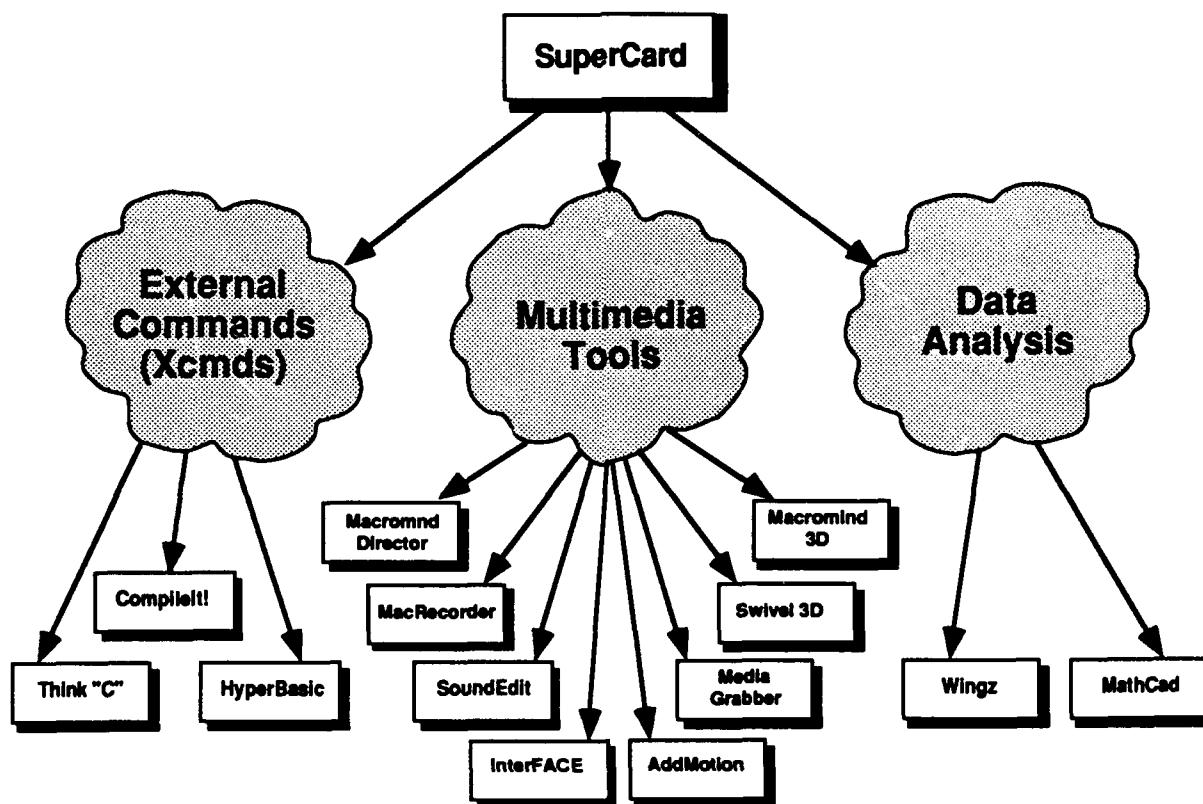


Figure 29. COTS-Based Architecture

By making extensive use of the COTS development tools, we were able to expedite the development phase of the Trainer and produce a complete prototype in a fraction of the time it would take using conventional development techniques. The COTS tools we used can be divided into three areas: Xcmd compilers, multimedia authoring environments, and data analysis tools. The 4th generation language (4GL) SuperCard and code links supplied by the COTS packages glued the tools together.

4.3.1 Hypermedia Authoring Environment

We used SuperCard, a hypertext-based, object-oriented prototyping and development language, as the main tool to develop this software. SuperCard, a 4GL similar to HyperCard, has expanded features such as sophisticated color objects, a powerful script editing environment, a symbolic debugger, and the ability to generate stand-alone applications. Figure 30 shows a sample SuperCard screen in the editor environment.

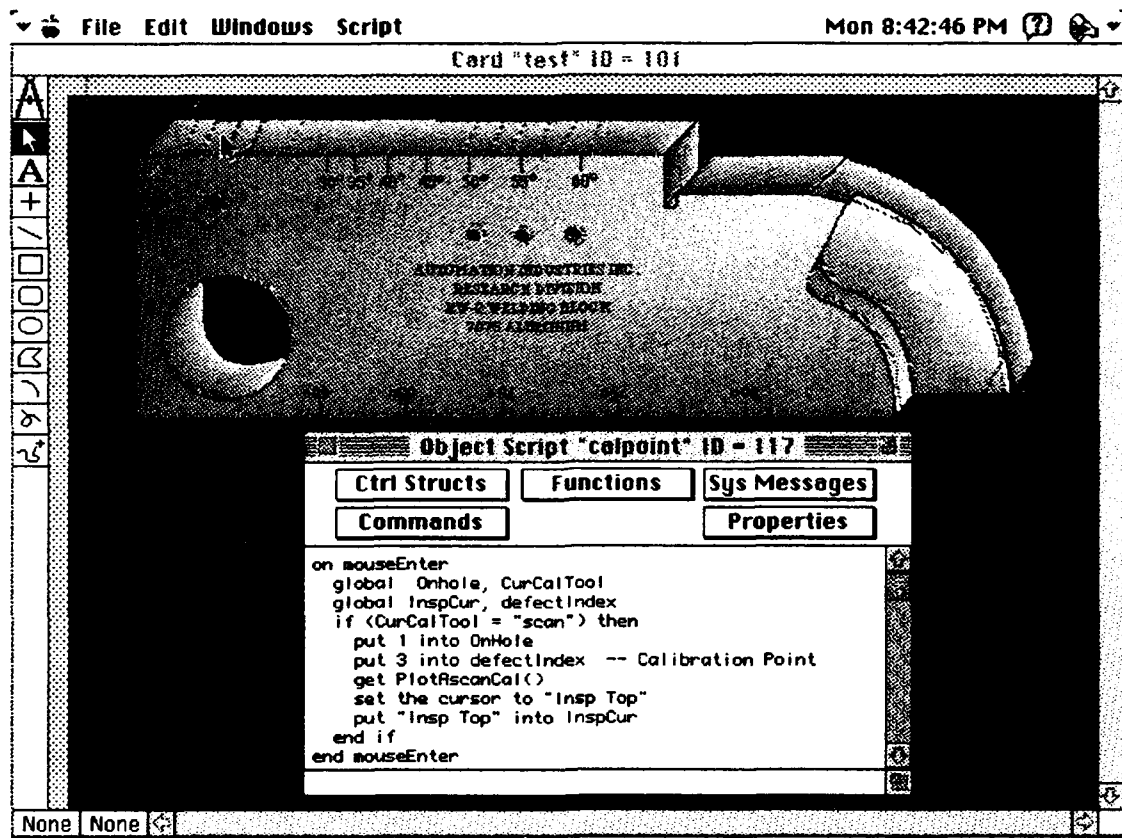


Figure 30. Sample SuperCard Display

SuperCard provides a complete development environment in which the designer creates objects that are operated on and linked by what is referred to as scripts. Scripts are written in an English-like proprietary language called SuperTalk.

Pull-down menus, at the top of the screen, provide access to SuperCard's main functions. At the left of the screen is a tool palette that is used to create and modify objects such as buttons, graphics, text fields, and bitmaps. The true power of SuperCard is its ability to create invisible objects and embed scripts within the objects to perform specific functions. The object scripts are invoked by system- or user-defined messages which are initiated by SuperCard.

Figure 30 displays a window that contains a script for a hidden object, which is located on the simulated "ALUMINUM" calibration block and is used to invoke a signal plotting routine for a standard calibration point. The script is executed when the mouse enters the object, thus changing the A-scan plot on the simulated ultrasonic meter. This technique of changing signals is extremely efficient because processing only takes place when the mouse enters an object. In a non-object-based system, the cursor position would have to be continually monitored and compared to the coordinated areas of signal change, wasting CPU cycles.

4.3.2 External Command Compilers

On the Macintosh, the concept of Xcmds addresses the issue of linking various COTS packages together. The Xcmds are external code resources which adhere to a standard calling convention and can be invoked from many 4GLs such as SuperCard. Although originally developed to provide a link to compiled functions, Xcmds are now used as gateways into third-party software. Many Xcmds are now available for integrating packages and providing expanded "add-in" functions. Furthermore, Xcmds can give software the ability to control hardware devices, perform complex animation sequences, and play computer-generated music. In the past, sample C or Assembler code would have been included in the manual, and the programmer would have had the cumbersome task of making it work. With the advent of Xcmds, programmers simply move the Xcmd resource to the application's resource fork and call it from the application.

Besides bridging the gap between various third-party software, Xcmds provide a method for developing time-critical code which can be called from a 4GL script. Typically, 4GLs are interpreted and have poor performance when compared to compiled code. The Xcmds allow the programmer to develop compiled functions that can be called from a script, thus increasing the execution speed of the software. The Trainer uses this method to develop time-critical, signal-generation routines.

The Trainer application incorporates data collected from field inspections into the Trainer software. By doing so, the student can evaluate "real" signals prior to performing an actual inspection. Because actual data is used, the signal-generation software has to be plotted in real time. This requires developing Xcmds that would apply filters to the signal. The main tool used for Xcmd development is Heizer Software's *CompileIt!*. Developing Xcmds using 3GLs

is a nontrivial task requiring a deep understanding of the Xcmd's calling conventions and parameter block. *CompileIt!* takes a novel approach to Xcmd programming by allowing Xcmds to be written in HyperScript, a language similar to a 4GL script. With this approach, *CompileIt!* takes advantage of the programmer's knowledge base and prevents learning a new language to develop Xcmds. Figure 31 is an example of a *CompileIt!* script.

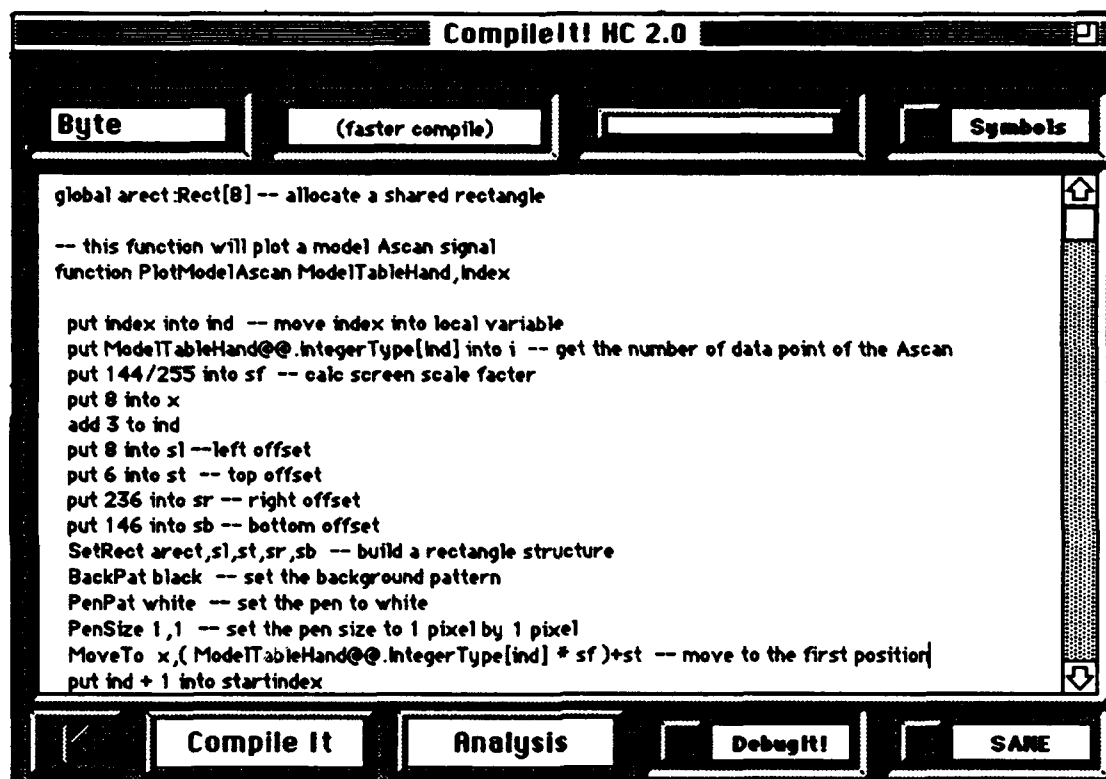


Figure 31. *CompileIt!* Development Environment

CompileIt! provides for advanced structures as well as full compatibility with the Macintosh Toolbox. *CompileIt!* also supplies tools to analyze program deficiencies. One of the slowest mechanisms in Xcmd execution is what is termed a "text callback," which occurs when the Xcmd needs the services of the calling program. When HyperCard's services are needed, such as checking if the mouse is within an object, the Xcmd passes the "mousewithin" query back to HyperCard. Then, HyperCard answers with a True or False. In a text callback, many conversions must take place, resulting in slow processing. *CompileIt!* allows the user to analyze callbacks and points out areas that can be optimized.

Another feature unique to *CompileIt!* is its sophisticated optional debugging facility. This utility attaches a symbolic debugger to the application's code resource that is activated upon entering the Xcmd. Figure 32 is a sample screen of *CompileIt!*'s debugger. Within the debugger, the programmer can set and clear breakpoints, view and modify containers (variables), and review callbacks. In short, *CompileIt!* has opened Xcmd development to the novice Macintosh developer.

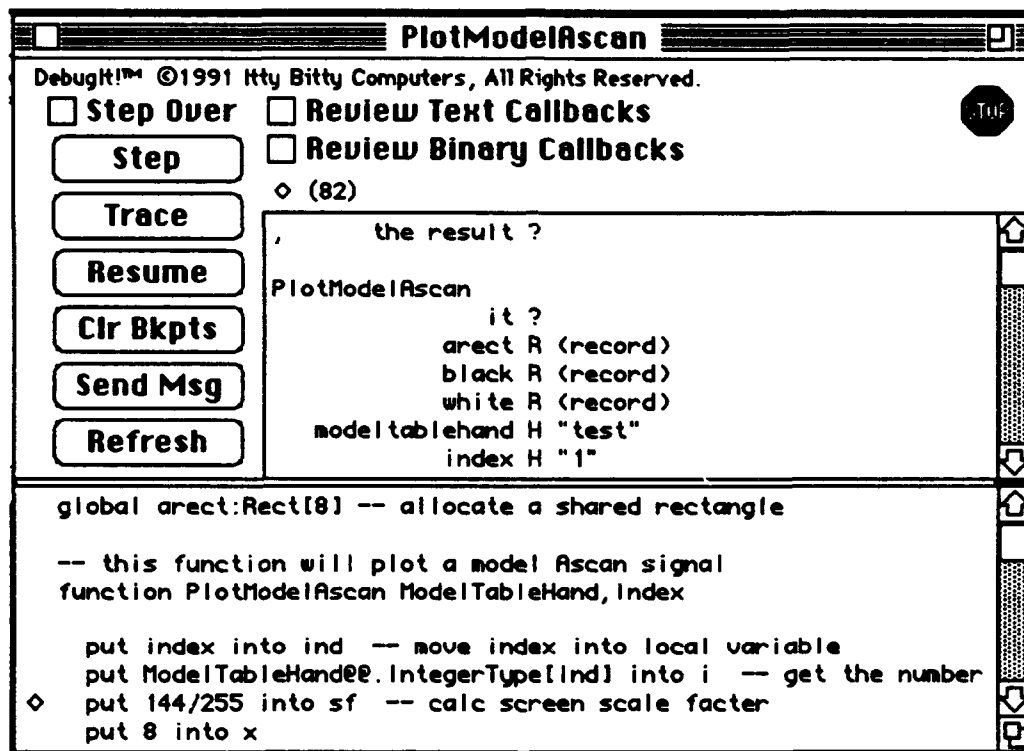


Figure 32. *CompileIt!* Debugger

4.3.3 Multimedia Development Tools

4.3.3.1 Digitized Voice

To make the Trainer software captivating and user-friendly, we made extensive use of Farallon Computing's *MacRecorder* and *SoundEdit* to create digitized sound. *MacRecorder*, an external hardware device used to capture digitized sound, is accompanied with a sound editor and an Xcmd utility for playing the sound via a 4GL. Figure 33 shows the *SoundEdit* screen which allows the developer to record and manipulate the sound. *SoundEdit* also provides a wide

array of effects (filters) that can be applied to the sound. For example, an echo filter can be applied to the voice sample to create an echo chamber effect. The developer can also rearrange words by simply cutting and pasting the sound spectrum.

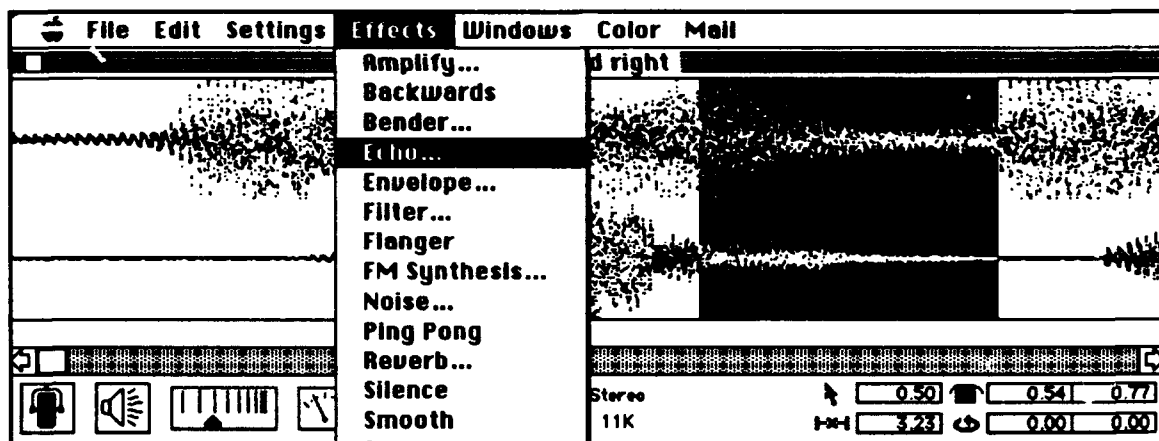


Figure 33. Sample MacRecorder Display

Once the sound has been defined, it is saved as a “snd resource” and imported into SuperCard via a menu selection. Next, an Xcmd for playing sound is imported into SuperCard. To play the sound, the programmer simply enters the script: “PLAYSND INTRODUCTION.” The number of sounds is limited only by the amount of available memory; but caution must be used, because digitized sound requires huge amounts of space.

4.3.3.2 Context-Sensitive Help System

In keeping with the objective of providing a captivating interface for novice computer users, we used a package called Bright Star Technology’s *InterFACE* (Interactive Facial Animation Construction Environment) to develop an on-line help system. *InterFACE* provides tools to create on-screen talking agents that interact with and guide users through the training system. The *InterFACE* package consists of two major products: a complete graphical editing environment that assists developers in creating the customized on-screen agent, and voice synchronize. Agent images can be drawn with these supplied tools or created by importing digitized images captured with a video camera or scanning device. To create a new agent, the developer draws or captures the agent’s face with the mouth position resembling the supplied template. Figure 34 shows the agent editor with the agent’s mouth position saying the letter

“F.” For each letter or sound, the developer must draw the agent with the correct mouth position. To produce reasonable, simulated, mouth-to-voice synchronization, a total of 32 agent images must be created.

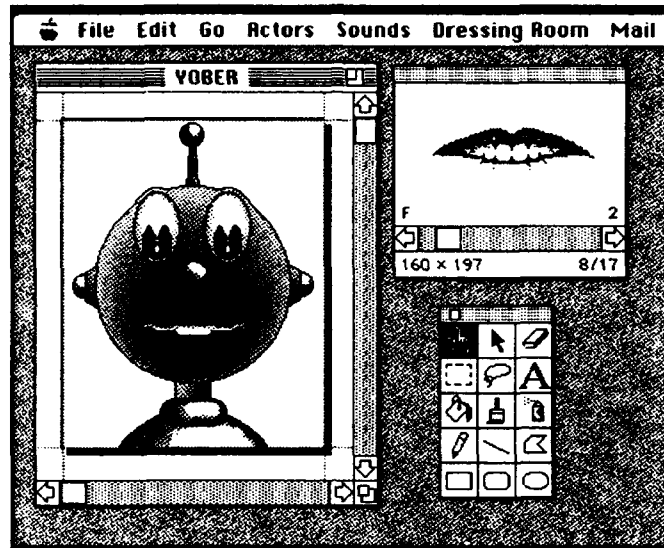


Figure 34. *InterFACE* Agent Editor

Once all the agent's expressions have been drawn, the developer may then synchronize the agent's facial movement with digitized voice. To do this, the developer simply imports the digitized voice segment and types in the spoken sentence. *InterFACE* converts the text to phonemes and then to its internal language, which consists of phonetic symbols plus facial expression codes. Finally, the user must test the end result and perform timing adjustments as necessary. If digitized voice is not required, the *Macintalk* voice synthesizer software could be used. Although the *Macintalk* driver software does not produce the same degree of definition as digitized voice, it uses far less memory and requires little or no synchronization.

InterFACE may also be called from traditional programming languages such as C and Pascal or from 4GLs. A driver, called RAVE, is placed in the system folder of the Macintosh and is activated via low-level calls. In SuperCard, the supplied Xcmd is called to activate the talking agent. For example, to wake the agent and have it say, "Hello can I help you," the following command would be invoked:

RAVE{Hello Can I Help You.}.

The RAVE Xcmd will then call the RAVE driver, which will control the computer-generated voice and synchronization of facial expressions. Another useful feature of the package is the ability to have multiple agents interacting on the screen simultaneously. Thus, agents can be programmed to talk with each other during tutorials or informational sessions.

4.3.3.3 Real-Time Digitized Video

Real-time digitized video is used to create tutorials for the Trainer software. The tutorial shows proper scanning procedures and test equipment calibration steps. To create real-time digitized video, a special video card and supporting software must be used. We used the RasterOps' 24XLTV card. This card supports 24-bit color and has the ability to capture real-time video from video cameras and VCRs. The card also comes with *MediaGrabber* software for capturing frames of video to secondary storage. Figure 35 is a sample screen of the *MediaGrabber* utility.

In the example screen, 10 frames of video are being captured at the rate of 1 frame per second. Depending on the bit depth of the image, up to 15 frames per second can be captured. (The standard frames per second of a VCR or video camera is 30.) Compression boards are slowly becoming available to capture true real-time video, but we have achieved a high degree of realism displaying 10 frames per second.

Once the video frames have been captured on disk, *Director* combines all captured frames into a video sequence. Overlaid titles and sound can then be added to the video sequence to create the desired effect. The combination of digitized video segments, sound, and titles is then saved as *Director* "movies," which can be played through an Xcmd movie player supplied with *Director*.

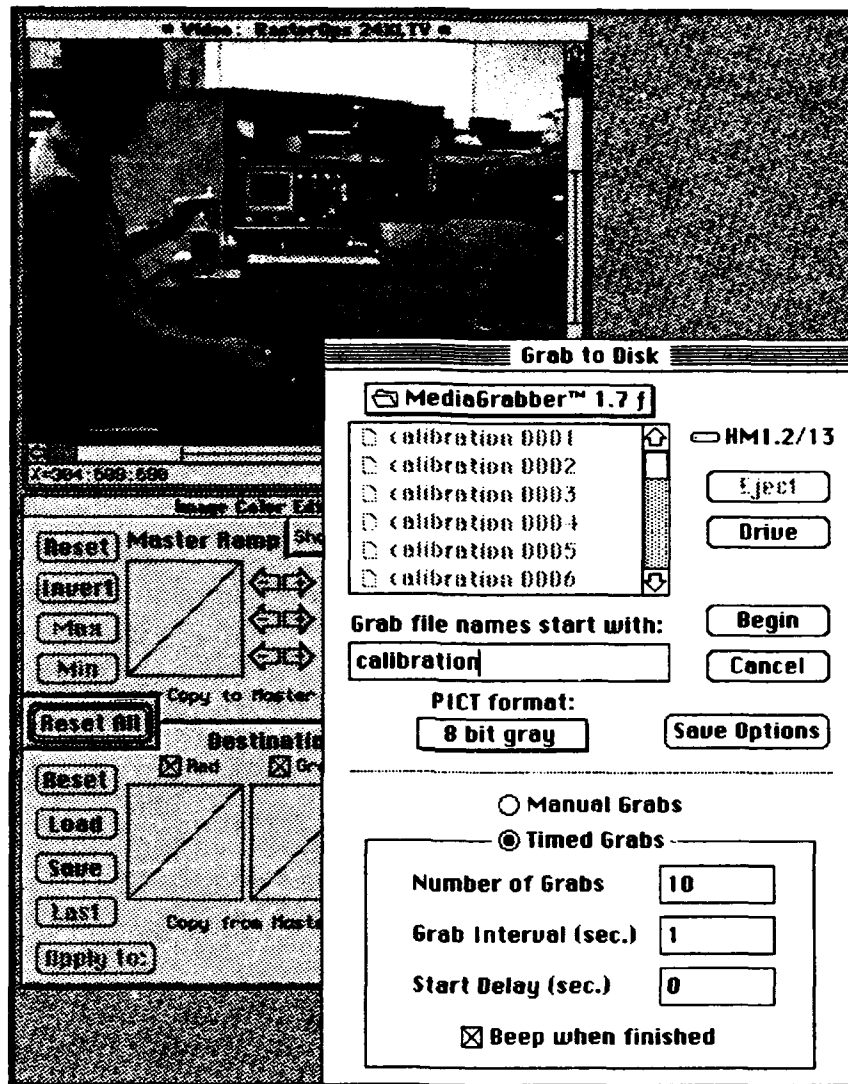


Figure 35. MediaGrabber Development Environment

SECTION 5

OTHER APPLICATIONS

During the development of the Trainer, different user organizations reviewed and commented on the prototyping. As explained previously, SM-ALC NDI personnel have worked closely with us throughout the development. Their comments have helped focus the Milstar prototyping and indicated other training applications where multimedia technology could provide similar benefits. We have also had the opportunity to present this work at American Society for Nondestructive Testing (ASNT) conferences ([3] and [4]), to the Air Force Wide NDI Meeting, and to various ALCs. Based on feedback received at these meetings and conferences, several enhancement areas have been identified and are summarized below.

The Trainer has been prototyped for pulse-echo ultrasonic inspection of Milstar radomes on C-135 aircraft. The most important consideration for enhancements are those related to Milstar. First, we considered the impact that different size and shape radomes (for different platforms) would have on the Trainer. This would only require changes in the radome graphics; the software would require little, if any, changes. Second, we considered the impact of a new composite for Milstar Radomes. Since signal data is read from data files into the Trainer, most of the updates would require that designers capture new reference signals from inspections and import them into the Trainer. At best, this would require no software changes. However, if the size of A-scans differed significantly from existing A-scans, some tables might have to be resized and the signal generation code modified slightly. In addition, the image tutorials are based on the Kevlar/polyester composite. A new composite would necessitate acquiring new images to import into the Trainer. Since there is no designer utility for making these changes, the updates would have to be made in the SuperCard environment. Still, the changes would require very little software modifications. Finally, a different calibration standard would require only updates to the calibration display in terms of the graphical objects that are displayed and invisible (for sensing transducer position). These and other updates only require small, isolated changes, because the application is structured in terms of objects that are relatively independent.

Another straightforward enhancement to the Trainer would be reusing many of the functions to provide an inspection tool that could capture inspection results. We anticipate that many areas of the Trainer, such as the radome graphics, drawing tools, tutorials, and ultrasonic signal database, would be reused. An inspection tool would allow inspectors to record their results electronically for later recall and review. Each inspection would record a radome map with the flawed areas outlined and classified. A log of inspections could be provided for each radome, and analysis functions could allow designers to quickly compare the results from one inspection to the next.

A key aspect of the Trainer is the multimedia technology for on-line help and tutorials. Ideally, experienced NDI personnel at ALCs could use this technology to create custom tutorials for new inspections, advanced inspections, or general tutorials. In cases of new and advanced inspections, often training procedures do not exist. In some cases, written procedures are generated and sent out into the field. In many cases, an ALC representative will be sent to several sites to provide training. A popular example has been the use of video tape generated at

an ALC and sent out to field sites. Sometimes ALC representatives are sent out to the field to provide training on existing inspection techniques that are not being performed adequately. Using multimedia technology, a tool could be prototyped that would allow ALCs (designer users) to create their own tutorials. The tutorials would be based on signals, images, digitized video frames, and voice. The designer would acquire the signals and images, create video frames and voice overlays, and use an environment that allows the different media to be integrated and either played (like a browser) or used for testing (show several images and have the student identify the flaw types). Interaction would be provided by allowing the designer to specify how areas of images, signals, etc., should react to mouse position or selection. Tutorials could be sent out as diskettes, or downloaded via modem, to the sites that require the specific training.

One of the significant benefits of the Trainer is the ability for student inspectors to practice on electronic test parts before evaluating production parts. Using existing technology in the Trainer, different signals and test parts could be captured and manipulated electronically. The changes required for ultrasonics using different test parts would be relatively straightforward. Different test parts, such as wing flaps, would require new graphics, signal data, and tutorials. We envision that a library (database) of electronic test parts could be generated, maintained, and provided to multiple training sites. Similar benefits could be derived for training in eddy current, where signal interpretation is critical and requires extensive training.

SECTION 6

CONCLUSION

We have described the background, functions, and development approach of the Milstar Radome NDI Trainer Prototype. At the time of this writing, the Trainer design had been completed and most functions had been prototyped. We anticipate that the Trainer will be completed in early FY93. A preliminary delivery to SAC field inspectors will occur in late FY92.

When this work started, there was little or no documentation on the application of multimedia technology to NDI training. This work has been highly experimental and has investigated how images, interactive signals, digitized video, and voice can address NDI training problems. The Trainer has addressed the particular issues associated with inspecting Milstar radomes. Obviously, as noted in section 5, the techniques apply to other NDI training problems. A close working relationship with SM-ALC NDI personnel has helped the prototyping to focus on areas that are most problematic. This work has illustrated how ESD and MITRE can apply prototyping to tackle a specific project application and advance the state of the art.

LIST OF REFERENCES

1. Air Force Technical Manual, 1 March 1992, *Nondestructive Methods*, Published under authority of The Secretary of The Air Force.
2. Boehm, B., 1985, *A Spiral Model of Software Development and Enhancement*, Wang Institute Presentation, Tyngsboro, MA.
3. Bailey, D., and Wilson, J., *Application of Multimedia Technology to NDE Training and Education*, Proceedings of The American Society for Nondestructive Testing (ASNT) Spring Conference, 30 March-3 April 1992, Orlando, FL.
4. Wilson, J., and Phair, D., *An Automated Trainer for Ultrasonic Inspection of Aircraft Radomes*, Proceedings of The American Society for Nondestructive Testing (ASNT) Fall Conference, 15-18 September 1991, Boston, MA.

GLOSSARY

3GL (third generation language): A procedural language, such as Pascal, C, and Fortran.

4GL (fourth generation language): A modern programming language that contains both procedural and nonprocedural features, such as SuperTalk and C++.

ALC (Air Logistics Center): Supports specific aircraft components, inspections, and repairs.

A-Scan: A trace that shows amplitude over time, from the initial ultrasonic pulse until the BSE is received. Most ultrasonic field inspections use A-scans.

BSE (back surface echo): The reflection of the ultrasonic pulse off the back wall (inside surface) of the test part, such as a radome.

COTS (commercial off-the-shelf): Refers to the use of commercial hardware and software for application development, as opposed to custom development.

CPU (central processing unit): The computer control logic used to execute programs.

Eddy Current: Testing that measures the magnitude and direction of electrical currents in relation to the part under test.

FSE (front surface echo): The reflection of the ultrasonic pulse off the front surface of the radome.

Multimedia: The integration of graphics, video, voice, animation, imagery, and sound into an application.

NDI (nondestructive inspection): Consists of inspection methods that do not require modifying the part under test, such as ultrasonics, X-ray, and eddy current.

RF (radio frequency): Refers to frequencies that can be generated by radios.

ROM (read-only memory): A nonvolatile storage device that holds fixed programs and data, such as the Macintosh system toolbox.

SEM (scanning electron microscope): An instrument that can acquire microscope images of a part under inspection.

SM-ALC (Sacramento Air Logistics Center): Supports ESD through the Milstar Program and has provided the domain expertise for the Radome Inspection Trainer.

Xcmd (external command): A code resource, such as a compiled function, that can be attached to a SuperCard Project.